SUNDB数据库管理系统 V5.0-入门指南

Getting Started

目录

1.入门 1

1.1 前言 1

1.2 概要 2

1.3 SUNDB 集群的特征 6

2. 指南 13

2.1 SUNDB实例管理 13

2.2 SUNDB安装删除与创建数据库 22

2.3 数据库存储结构管理 55

2.4 模式对象(Schema Object)管理 67

2.5 用户管理 71

2.6 SUNDB参数 77

2.7 SUNDB实用程序 79

3. Cluster指南 85

3.1 SUNDB集群系统管理 86

3.2 安装SUNDB与创建数据库 98

3.3 管理模式对象 105

3.4 SUNDB参数 122

4. What's New 127

4.1 特征矩阵 127

4.2 SUNDB V5.0 22的新功能 262

4.3 补丁说明(Patch Notes) 273

1.入门

1.1 前言

本用户手册是为配置管理和操作SUNDB的运营人员而编写的指南本手册的目的在于介绍安装及管理SUNDB时所需的基本概念以及描述使用SUNDB系统时的注意事项

- 本文档中所提供的描述不具备绝对性可根据安装环境和使用方法发生变更
- 本文档的编写以SUNDB V5.0 22版本为准
- 本文档的编写以RedHat系列Linux平台为准

目标读者

本文档的目标读者如下

- 基于SUNDB 数据库开发程序并需要了解相关基本管理方法的人员
- SUNDB 数据库的使用者及性能管理员
- SUNDB 集群系统的使用者及性能管理员

1.2 概要

本章为初次使用SUNDB的用户介绍SUNDB的基本架构及特点用户可以选择使用SUNDB单机版或集群版中的一个以下分别说明其架构和使用方法的区别

SUNDB 数据库管理系统

SUNDB数据库系统构成如下

- 用户安装的SUNDB二进制软件包
- 为由多个共享内存组成的表空间的集合的数据库
- 支持数据库持久性的各种文件
  - 每个共享内存创建一个与共享内存大小相同的数据文件
  - 发生故障时可恢复数据库的重做日志文件
  - 用于设置数据库的各种配置文件
  - 其他记录数据库运行中发生的事件等内容的跟踪日志文件
- 管理数据库的gmaster进程及其内部的多个系统线程

SUNDB 架构

SUNDB数据库为了防止特定应用程序引起的错误扩散到整个数据库不采用多线程架构而采用了基于共享内存的多进程架构SUNDB数据库的整体架构如[图-1]所示在共享内存上加载数据

gmaster进程是一个管理守护进程管理boot-up, log flush, aging等整体数据库操作此外在磁盘上

记录重做日志文件和数据文件以保障数据的持久性使用SUNDB数据库的应用程序可使用以下两种访问模式中的一个

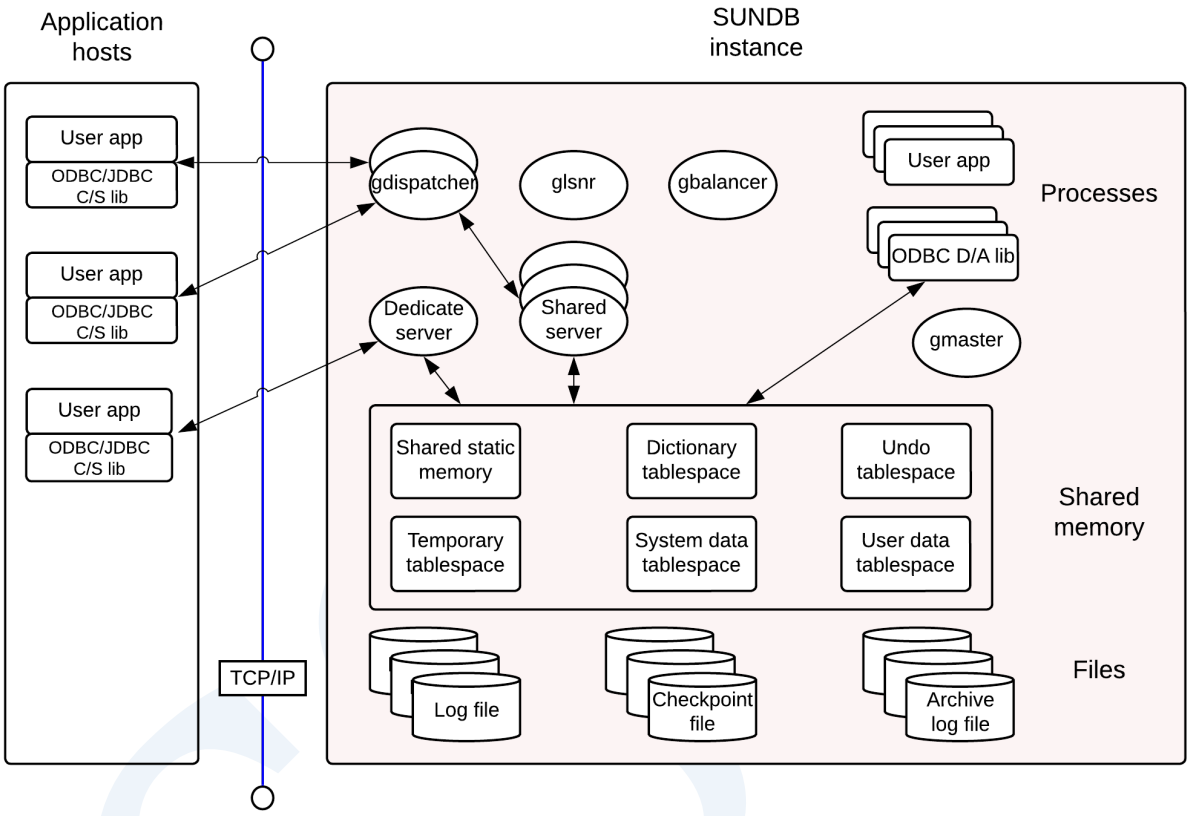


Figure 0-1 SUNDB architecture

- Direct Access (D/A) model
  - 用户应用程序与SUNDB数据库在同一台服务器上运行时可使用
  - 用户应用程序需要链接D/A专用SUNDB ODBC/JDBC库后才可使用
  - D/A专用SUNDB ODBC/JDBC库内包含语句处理和存储管理模块可以直接访问构成数据库的共享内存来处理用户请求
  - 可消除应用程序进程与数据库处理模块之间的通信负载适合实现部分需要低延迟的业务
- Client/ Server (C/S) model

- 用户应用程序与SUNDB数据库在同一台或不同服务器上运行时可使用
- 用户应用程序需要链接C/S专用SUNDB ODBC/JDBC库后才可使用
- C/S专用SUNDB开发库通过和连接字符串中指定的服务数据库的服务器（gserver）的TCP通信处理用户的请求
- 虽然单一应用程序的响应速度低于D/A模式但其不依赖于应用程序的位置即使应用程序出现故障也能较稳定的运行
- C/S model分为专用模式和共享模式专用模式是一个客户端对应一个服务器（gserver）进程的结构共享模式是调度程序(gdispatcher)与共享服务器(gserver)一直在运行并应对多个客户端的方式
- 专用模式适合数据量大的业务共享模式适合客户端数量多但数据量不太大的业务
- 专用，共享模式的设置参考**odbc.ini文件**, **Listener Configuration**

> Note:
>
> D/A模式下应用程序直接访问数据库进行操作因此在开发初期会产生较多问题而导致数据库实例不稳定所以在开发初期阶段首先用C/S模式进行开发后在最后阶段转换为D/A模式会更加有效

SUNDB 集群系统架构

SUNDB可以配置成单机版(Standalone)数据库使用也可以将多个数据库绑定为集群（Cluster）以集群为单个管理单位来使用通过构成集群系统使用时用户可以根据所需的分片策略将表数据分发并存储到多个节点中因此在服务的高可用性与并行处理方面有很大优势

SUNDB集群系统保障集群广泛（cluster wide）执行的事务的ACID因此当连接到属于集群系统的

任何一个节点执行事务它提供的数据可靠性与在单机版服务器上执行事务时的数据可靠性相同

属于SUNDB集群系统的每个数据库在多进程结构与数据加载方法等方面大部分与单机版数据库的架构相同只是其添加了用于集群中的成员节点之间进行高效通信的进程（cdispatcher）与用于在集群成员节点上进行数据存储及管理的集群服务器（cserver）进程共享存储中添加了用于管理集群系统的事务的表空间和管理区域

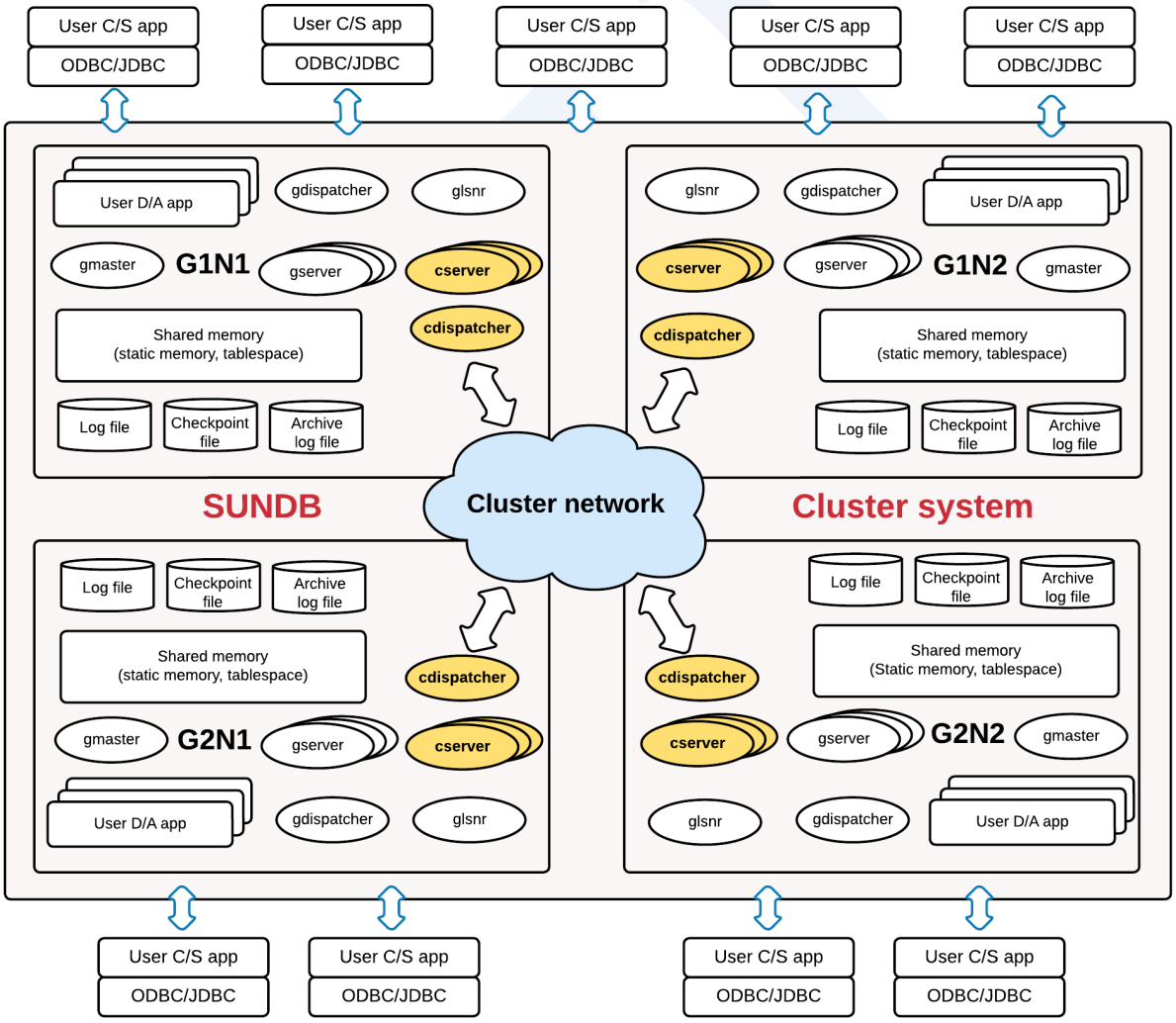


Figure 0-2 SUNDB cluster system architecture

1.3 SUNDB 集群的特征

集群的特点

SUNDB集群是克服了现有单机版系统的事务性能局限与存储空间局限的无共享结构的集群系统

- 高性能(High Throughput)
  - SUNDB集群可无限制的添加组(group)并且可以通过添加组来线性提高性能
  - 可克服现有的基于内存的单机版系统存储空间的局限性
- 高可用性(High Availability)
  - 一个组由多个成员构成时如果组内至少有一个成员在运行则不会影响可用性
  - 即使一个组中的所有成员都不可用时除该组之外的其他组也可以正常提供服务
- 在线扩展及在线恢复
  - 在正在进行服务的状态下也可以添加组或成员并不会对正在进行中的服务产生影响
  - 因故障导致服务被中断的成员也可以在线参与集群
- 提供完美的事务
  - 完美提供事务应遵循的下列属性
    - 原子性(Atomicity)
    - 一致性(Consistency)
    - 隔离性(Isolation)
    - 持久性(Durability)
- 提供完美的多版本并发控制(MVCC multi-version concurrency control)
  - SUNDB集群提供与单机版系统相同的Global Statement Level Consistency
  - 从特定点开始的SQL访问任何节点时均可在多个版本中访问任何一个所需点的版本

- 提供标准SQL及标准DBC
  - 提供SQL 92为准的SQL
  - 提供JDBC/ODBC等各种标准DBC
- 应用程序的兼容性
  - 不需要修改在现有单机版系统中开发的应用程序源或SQL可直接在SUNDB集群中使用

集群约束

除以下约束外SUNDB集群均可使用与单机版相同的SQL语句

> Note:
>
> Sharded table的主键(PRIMARY KEY)UNIQUE约束条件UNIQUE INDEX应包含分片键(sharding key)

以下示例是由于UNIQUE(name)约束条件未包含作为分片键的id列而失败的情况

```
gSQL>
CREATE TABLE t1
(
    id   INTEGER PRIMARY KEY,
    name VARCHAR(128),
    UNIQUE (name)
)
SHARDING BY HASH(id);
```

```
ERR-HYC00(16380): UNIQUE or PRIMARY KEY must include all sharding key
columns for cluster system
```

如下应当生成包含UNIQUE（idname）或UNIQUE（nameid）等分片键的约束条件

```
gSQL>
CREATE TABLE t1
(
    id   INTEGER PRIMARY KEY,
    name VARCHAR(128),
    UNIQUE( id, name )
)
SHARDING BY HASH (id);

Table created.
```

> Note:
> 非确定性（non-deterministic）语句应具有全局二级索引（global secondary index）以区分集群成员之间的相同的行

以下是省略全局二级索引后强制创建表时发生的错误的示例

```
gSQL> CREATE TABLE t1 ( c1 INTEGER ) WITHOUT GLOBAL SECONDARY INDEX;

Table created.
```

```
gSQL> INSERT INTO t1 VALUES (1), (2), (3), (4), (5);

5 rows created.

gSQL> COMMIT;

Commit complete.
```

以下是删除三条的示例语句无法保证集群成员删除相同的行

```
gSQL> DELETE FROM t1 FETCH 3;

ERR-42000(16423): does not support non-deterministic DML in the cluster
system : global secondary index expected
```

以下是无法保证集群成员使用RANDOM（1,100）将相同的行更改为相同的值的示例

```
gSQL> UPDATE t1 SET c1 = RANDOM(1, 100);

ERR-42000(16423): does not support non-deterministic DML in the cluster
system : global secondary index expected
```

以下是使用可更新游标（updatable cursor）更改当前位置的行的示例为了识别集群成员之间相同的行需要全局二级索引

```
gSQL> \var v1 INTEGER
```

```
gSQL> DECLARE cur1 CURSOR FOR SELECT c1 FROM t1 FOR UPDATE;


Cursor declared.


gSQL> OPEN cur1;


Cursor is open.


gSQL> FETCH cur1 INTO :v1;


V1
--
 1


1 row fetched.


gSQL> UPDATE t1 SET c1 = 1 WHERE CURRENT OF cur1;


ERR-42000(16423): does not support non-deterministic DML in the cluster
system : global secondary index expected
```

> Note:
>
> 不支持可延迟的约束条件

```
gSQL> ALTER TABLE t1 ADD CONSTRAINT t1_uk UNIQUE(id) DEFERRABLE;
```

```
ERR-HYC00(16388): does not support deferrable constraints in the cluster
system :
ALTER TABLE t1 ADD CONSTRAINT t1_uk UNIQUE(id) DEFERRABLE
                                               *
ERROR at line 1:
```

> Note:
>
> 在不同服务器中使用相同的序列时无法保证是否按照顺序使用

- 在g1n1服务器中执行

```
gSQL> SELECT seq1.NEXTVAL FROM dual;

NEXTVAL
-------
      1

1 row selected.

gSQL> SELECT seq1.NEXTVAL FROM dual;

NEXTVAL
-------
      2
```

```
1 row selected.
```

- 在g2n1服务器中执行

```
gSQL> SELECT seq1.NEXTVAL FROM dual;

NEXTVAL
-------
     21

1 row selected.
```

- 再次在g1n1服务器中执行

```
gSQL> SELECT seq1.NEXTVAL FROM dual;

NEXTVAL
-------
      3

1 row selected.
```

2.指南

2.1 SUNDB实例管理

本章介绍管理SUNDB实例的基础知识

概述

SUNDB数据库系统由数据库和实例组成数据库是启动数据库所需的各种文件的集合例如内存上的字典数据用户数据以及其数据文件在线重做日志文件等

数据库实例由两部分组成一个是包含用于运行SUNDB数据库的状态（run-time）信息的内存部分另一个是用于其操作和管理的后台进程每个数据库实例分为构成共享内存时使用的共享内存键值和“SUNDB_DATA”环境变量值

参数设置

$SUNDB_DATA/conf/sundb.property.conf文件记述了SUNDB的参数用户可使用默认参数安装SUNDB此章节仅说明TBS (Tablespace) , LOG, CONTROL FILE等主要参数

以下为安装SUNDB时的主要参数和对应说明

参数	说明	默认值
SYSTEM_TABLESPACE_DIR	存储系统TBS的目录路径该路径中安装了如下TBS • DICTIONARY_TBS • MEM_DATA_TBS • MEM_UNDO_TBS • MEM_TEMP_TBS • MEM_TRANS_TBS	'<SUNDB_DATA>/db'
SYSTEM_MEMORY_DICT_TABLESPACE_SIZE	字典表空间大小	128M
SYSTEM_MEMORY_DATA_TABLESPACE_SIZE	内存数据表空间大小	200M
SYSTEM_DISK_DATA_TABLESPACE_SIZE	磁盘数据表空间大小	200M
SYSTEM_MEMORY_UNDO_TABLESPACE_SIZE	undo表空间大小	32M
LOG_DIR	默认日志目录路径	'<SUNDB_DATA>/wal'
SYSTEM_LOGGER_DIR	系统日志目录路径	'<SUNDB_DATA>/trc'
CONTROL_FILE_COUNT	控制文件数量	2
CONTROL_FILE_0	第一个控制文件的路径	'<SUNDB_DATA>/wal/control_0.ctl'
CONTROL_FILE_1	第二个控制文件的路径	'<SUNDB_DATA>/wal/control_1.ctl'

参数	说明	默认值
BUFFER_CACHE_SIZE	用于缓存在磁盘表空间生成的表索引page的缓冲区缓存的大小	64M

Table 2-1 主要参数

Note:

* DICTIONARY_TBS：存储SUNDB的字典表的表空间

* MEM_DATA_TBS：创建用户表/索引的内存表空间

* DISK_DATA_TBS：创建用户表/索引的磁盘表空间

* MEM_UNDO_TBS：存储事务的Undo(rollback)信息的表空间

* MEM_TRANS_TBS (仅限集群)：在集群系统中用于恢复全局事务而使用的表空间大小根据事务表(transaction table)大小及集群节点数量自动计算

更改数据库或实例的设置时更改文本参数文件($SUNDB_DATA/conf/sundb.properties.conf)或在环境变量重新定义SUNDB_<property_name>形式的变量即可参数文件的设定值的优先级高于环境变量的设定值

后台进程

SUNDB拥有用于管理实例的后台进程(gmaster)gmaster内部由多个系统线程构成其内容如下

参数	说明
Main thread	启动/结束gmaster进程
Log archiving thread	切换在线重做日志文件时将之前的重做日志文件拷贝到指定位置并保管
Ager thread	整理被删除的模式对象所使用的资源
Page flusher thread	发生checkpoint时为了将脏页保存到磁盘向IO slave分配作业并控制
Log flusher thread	把数据库运行时积累在重做日志缓冲区的日志记录周期性的写入在线重做日志文件
Checkpoint thread	切换重做日志文件时 把脏页回写到磁盘上的数据文件和在线重做日志文件
Cleanup thread	整理异常结束的客户端所使用的资源并回滚对应事务
IO slave thread	执行checkpoint或datafile loading等与数据文件相关的所有磁盘IO
Process monitor thread	执行并监控balancer (gbalancer), dispatcher (gdispatcher), shared-server (gserver) 等进程当异常结束时重启相关进程
Cluster recover thread (仅限集群)	在集群系统恢复全局事务
Failover thread (仅限集群)	集群系统中特定节点或网络发生故障时负责成员的offline及重新选择coordinator等failover处理

Table 2-2 系统线程

客户端进程

Client/ Sever模式

使用Client/ Server(以下称C/S)模式的应用程序连接到等待访问请求的监听器(glsnr)后生成新的数据库服务进程(gserver)(dedicated 模式)并通过与此进程的TCP通信处理用户的请求所有操作都是通过进程之间的通信来执行操作因此在应用程序端产生的任何信号均不会对数据库的状态产生影响cleanup threads定期检查非正常结束的应用程序所使用的资源并进行返还

Direct Access模式

SUNDB支持Client/ Server(C/S)模式的同时还支持Direct Access(D/A)模式所有应用程序通过链接SUNDB提供的服务器库直接访问数据库和实例因此没有单独的服务进程只有应用程序进程

当通过D/A模式创建的应用程序进程被运行过程中产生的信号(SIGNAL)异常中断时该进程所使用的资源由库连接时设置的信号处理函数按照如下两个阶段整理

1. 该进程的信号处理在其使用的会话对象标记异常终止后结束
2. 与C/S模式相同一定时间后由gmaster的cleanup thread返还

D/A模式下应用程序进程直接访问数据库区域因此需要注意以下事项

- 在D/A模式下运行的应用程序进程收到SEGV等致命的信号而结束时需要释放库连接时登记的信号处理所使用的共享资源因此在应用程序内安装特定信号的handler时需在连接数据库

之前声明

- 强制结束应用程序进程时不要使用进程无法检测到信号的SIGKILL(kill -9)而要使用SIGTERMSIGQUIT或SIGUSR2等

实例的内存结构

数据库实例使用的内存大小取决于参数文件中指定的相关参数实例使用的共享内存主要分为静态(static)区域和表空间区域静态区域存储数据库运行时进程之间共同使用的实例的基本信息与各SessionStatement及Transaction信息Redo Log BufferDictionary Cache信息以及其他各种运行信息表空间领域由存储各个表空间内容的Page Frame与控制这些的Page Control Header(PCH)构成

应用程序进程的内存除了连接时附加的实例内存之外还包含以进程为单位共享的ODBC环境和多个ODBC句柄以及包含Bind信息等其他信息的堆内存区域

实例的开始与结束

为了启动SUNDB实例SHARED_MEMORY_STATIC_KEY参数值的设置要不同于其他实例之后可以使用gsql启动SUNDB实例如下所示 要以sysdba角色执行

如果要在C/S模型的专用（dedicated）模式下启动或结束SUNDB实例需要提前运行**listener**

无法在C/S模式的共享模式下启动或结束SUNDB实例

```
% gsql sys gliese --as sysdba
```

```
Connected to SUNDB Database.


gSQL>
```

SUNDB 实例分以下多个启动阶段

- NOMOUNT
  - 启动管理SUNDB实例的gmaster守护进程
- MOUNT
  - 通过$SUNDB_DATA环境变量读取参数以及用于恢复的控制文件
- OPEN
  - 从数据文件加载表空间内容后使用重做日志文件进行恢复创建No-Logging Index生成Dictionary Cache后等待客户端的连接

用以下gsql 命令启动SUNDB实例

```
gSQL> \startup nomount
Startup success


gSQL> alter system mount database;
System altered.


gSQL> alter system open database;
System altered.
```

如果要直接进入OPEN阶段可执行以下命令

```
gSQL> \startup open

Startup success
```

如果结束SUNDB实例gmaster管理守护进程也同时终止因此无法再连接或进行其他数据库操作

可以用以下4种方式结束SUNDB实例

- NORMAL
  - 阻止新会话的连接等待已连接的会话全部结束后执行Checkpoint并结束实例
- TRANSACTIONAL
  - 阻止新事务的开始等待执行中的事务全部结束后执行Checkpoint并结束实例
- IMMEDIATE
  - 阻止新的单位运算（与SUNDB数据库的通信单位例如: FETCH or EXECUTE等）等待当前执行中的所有单位运算结束后回滚所有事务并执行Checkpoint后结束实例
- ABORT
  - 忽视连接中的会话的状态直接结束gmaster退出实例

结束实例时以sysdba角色执行gsql并按照如下执行`\shutdown` 命令

```
% gsql sys gliese --as sysdba


Connected to SUNDB Database.


gSQL> \shutdown normal


Shutdown success
```

```
gSQL>
```

Listener开始与结束

为了在C/S环境下提供服务需要启动listener

如下启动Listener

```
% glsnr --start

Listener is started successfully.

%
```

如下结束listener

```
% glsnr --stop

Listener is stopped.

%
```

关于listener控制的详细说明请参考**glsnr**

关于启动和结束集群系统的方法请参考**启动与关闭集群系统**

2.2 SUNDB安装删除与创建数据库

本章介绍安装SUNDB软件与创建数据库的方法

概述

SUNDB软件通过名为sundb-<version_no>-<os_type>-<cpu_type>.tar.gz的压缩文件提供解压后即在对应位置生成所需的软件二进制文件与各种Sample以及基本的数据库目录结构并安装成功安装完成后生成对应安装包名的目录并在其目录下生成 "sundb_home"与"sundb_data"目录

- "sundb_home"目录
  - SUNDB Product的二进制home
  - 定义为"SUNDB_HOME"环境变量
  - 存储二进制运行文件客户端链接库头文件许可等
- "sundb_data"目录
  - SUNDB生成的用户数据库（实例）的home
  - 定义为"SUNDB_DATA"环境变量
  - 存储数据文件日志文件参数文件等的默认位置

之后通过$ SUNDB_HOME / bin目录下的gcreatedb工具在$ SUNDB _DATA目录下创建数据库

软件支持平台

SUNDB可在以下版本的平台中使用

平台	平台名称	OS	CPU	备注
Server platform	linux-x86_64	linux	x86_64	>= linux kernel 2.6 >= glibc 2.1[1] >= gcc 4.1.2 >= java 1.6 <= java 1.8
	linux-powerpc-64	linux	powerpc	>= linux kernel 2.6 >= glibc 2.1[1] >= gcc 4.1.2 >= java 1.6 <= java 1.8
	hpux11.31-itanium-64	HP-UX 11.31	itanium	>= java 1.6 <= java 1.8
	aix7-powerpc-64	AIX 7.2	powerpc	>= java 1.6 <= java 1.8

平台	平台名称	OS	CPU	备注
Client platform	linux-x86_64	linux	x86_64	>= linux kernel 2.6 >= glibc 2.1[1] >= gcc 4.1.2 >= java 1.6 <= java 1.8
	linux-x86_32	linux	x86_32	>= Linux kernel 2.6 >= glibc 2.1[1] >= gcc 4.1.2 >= java 1.6 <= java 1.8
	hpux11.31-itanium-64	HP-UX 11.31	itanium	>= java 1.6 <= java 1.8
	hpux11.31-itanium-32	HP-UX 11.31	itanium	>= java 1.6 <= java 1.8
	aix7-powerpc-64	AIX 7.2	powerpc	>= java 1.6 <= java 1.8

平台	平台名称	OS	CPU	备注
	linux-powerpc-64	linux	powerpc	>= Linux kernel 2.6 >= glibc 2.1[1] >= gcc 4.1.2 >= java 1.6 <= java 1.8
	windows-x86-64	Windows	PENTINUM x86	>= java 1.6 <= java 1.8
	windows-x86-32	Windows	PENTINUM x86	>= java 1.6 <= java 1.8

Table 2-3 支持平台

[1]在CentOS 8RHEL 8以上用户要增加安装libnsl

系统要求事项

安装SUNDB之前需要确认以下系统要求事项

- 至少确保2G以上的物理内存空间与磁盘空间
- 确保充分的Paging(swap)空间
- 需要拥有符合要安装的平台的安装包版本

SUNDB安装包配置

本章介绍安装SUNDB时的目录结构

软件安装包目录结构

目录名称	服务器	客户端	说明
SUNDB_HOME	O	O	安装二进制文件与库更新时可覆盖（overwrite）的组
SUNDB_DATA	O	X	存储数据的实际路径无法覆盖（overwrite）的组

Table 2-4 上层目录结构

区分	名称	内容
SUNDB_HOME	admin	创建数据库时所需的模式脚本
	bin	执行文件
	lib	库文件
	include	ODBC, XA, Embedded SQL等头文件
	license	许可文件
	sample	sample文件
	msg	报错消息文件
	script	便利性脚本文件（预计后续支持）

区分	名称	内容
	app_dev	Application Development
SUNDB_DATA	conf	环境设置文件
	db	数据库文件
	wal	日志文件控制文件
	archive_log	归档日志文件
	backup	备份文件
	trc	追踪日志文件警告信息文件
	journal	用于集群再平衡的journal文件
	data	运行clustone时需要的TxFile

Table 2-5 安装包目录结构

安装包文件目录

本节介绍各目录包含的文件以及是否包含在服务器端安装包或客户端安装包中

文件名	服务器	客户端	说明
README	O	X	read me
DictionarySchema.sql	O	X	Dictionary Schema生成脚本
InformationSchema.sql	O	X	Information Schema生成脚本

文件名	服务器	客户端	说明
PerformanceViewSchema.sql	O	X	PerformancView Schema生成脚本

Table 2-6 admin/standalone目录

文件名	服务器	客户端	说明
README	O	X	read me
DictionarySchema.sql	O	X	Dictionary Schema生成脚本
InformationSchema.sql	O	X	Information Schema生成脚本
PerformanceViewSchema.sql	O	X	PerformancView Schema生成脚本

Table 2-7 admin/cluster目录

使用单机版SUNDB时使用admin/standalone目录中的生成脚本使用集群系统SUNDB时使用admin/cluster目录中的生成脚本

文件名	服务器	客户端	说明
README	O	O	read me
gmaster	O	X	SUNDB Master
gcreatedb	O	X	数据库创建工具
glsnr	O	X	Listener控制工具
gbalancer	O	X	用于C/S共享的负载均衡器

文件名	服务器	客户端	说明
gdispatcher	O	X	管理C/S共享的多个连接
gserver	O	X	C/S的实例管理器
gsql	O	X	交互式SQL工具
gsqlnet	O	O	用于C/S的交互式SQL工具
gpec	O	O	嵌入式SQL预编译器
logmirror	O	X	重做日志复制工具
cyclone	O	X	CDC复制工具
gloader	O	X	导入/导出工具
gloadernet	O	O	C/S的导入/导出工具
cymon	O	X	CDC监测工具
gdump	O	X	控制/日志/数据/二进制属性查看器
gsyncher	O	X	共享内存日志和磁盘日志文件的同步实用程序
tablediff.jar	O	X	表比较工具
cdispatcher	O	X	管理集群连接并在集群系统中分发协议
cserver	O	X	集群系统的实例管理器
gtrclogger	O	X	集群系统的跟踪日志管理器

文件名	服务器	客户端	说明
gmon	O	X	过程监控工具
galocator	O	X	位置管理工具
gagent	O	X	位置提供工具
gloctl	O	O	交互式位置编辑工具
cyfile	O	X	CDC导出文件工具
clustone	O	X	集群CDC复制工具
clustone_sender	O	X	集群CDC复制工具(sender tool)

Table 2-8 bin目录（Unix）

文件名	服务器	客户端	说明
README	X	O	read me
gloadernet.exe	X	O	C/S的导入/导出工具
gpec.exe	X	O	嵌入式SQL预编译器
gsqlnet.exe	X	O	用于C/S的交互式SQL工具
gloctl.exe	X	O	-

Table 2-9 bin目录(Windows client)

文件名	服务器	客户端 (64bit)	客户端 (32bit)	说明
README	O	O	O	read me
libstib.so	O	X	X	Infiniband用共享库
libsundb.a	O	X	X	ODBC的包含D/A和C/S的静态库
libsundba.a	O	X	X	ODBC的D/A专用静态库
libsundbas.so	O	X	X	ODBC的D/A专用共享库
libsundbc.a	O	O	O	ODBC的C/S专用静态库
libsundbcs-ul32.so	O	O	X	ODBC的64bit-C/S专用共享库 (SQLLEN = 4 byte)
libsundbcs-ul64.so	O	O	X	ODBC的64bit-C/S专用共享库 (SQLLEN = 8 byte)
libsundbcs.so	X	X	O	ODBC的32bit-C/S专用共享库
libsundbcvtGB18030_32.so	X	X	O	32bit GB18030字符集转换库
libsundbcvtGB18030_64.so	O	O	X	64bit GB18030字符集转换库
libsundbcvtUHC_32.so	X	X	O	32bit UHC字符集转换库
libsundbcvtUHC_64.so	O	O	X	64bit UHC字符集转换库
libsundbesql.a	O	O	O	嵌入式SQL用静态库
libsundbesqls.so	O	O	O	嵌入式SQL用共享库

文件名	服务器	客户端 (64bit)	客户端 (32bit)	说明
libsundbs.so	O	X	X	ODBC的包含D/A和C/S的共享库
sundb6.jar	O	O	O	C/S专用JDBC库(java 1.6用)
sundb7.jar	O	O	O	C/S专用JDBC库(java 1.7用)
sundb8.jar	O	O	O	C/S专用JDBC库(java 1.8用)
libsundbjni.so	O	X	X	D/A专用JDBC共享库
libsundbjnigc.so	O	O	O	全局连接专用JDBC共享库
libgdlc.a	O	O	O	ODBC的C/S专用静态库
libgdlcs.so	O	O	O	ODBC的C/S专用共享库

Table 2-10 lib目录（Unix）

文件名	服务器	客户端 (64bit)	客户端 (32bit)	说明
sundbc.lib	X	O	O	ODBC的C/S专用静态库
sundbcs.dll	X	X	O	ODBC的32bit-C/S专用共享库
sundbcs-ul64.dll	X	O	X	ODBC的64bit-C/S专用共享库(SQLLEN = 8byte)

文件名	服务器	客户端 (64bit)	客户端 (32bit)	说明
sundbsetup32.dll	X	X	O	ODBC的32bit setup library
sundbsetup64.dll	X	O	X	ODBC的64bit setup library
sundbesql.lib	X	O	O	嵌入式SQL的静态库
sundbesqls.dll	X	O	O	嵌入式SQL的共享库
sundbcvtGB18030_32.dll	X	X	O	32bit GB18030字符集转换库
sundbcvtGB18030_64.dll	X	O	X	64bit GB18030字符集转换库
sundbcvtUHC_32.dll	X	X	O	32bit UHC字符集转换库
sundbcvtUHC_64.dll	X	O	X	64bit UHC字符集转换库
sundb6.jar	X	O	O	C/S专用JDBC library (java 1.6用)
sundb7.jar	X	O	O	C/S专用JDBC library (java 1.7用)
sundb8.jar	X	O	O	C/S专用JDBC library (java 1.8用)
gdlc.lib	X	O	O	ODBC的C/S专用静态库
gdlcs.lib	X	O	O	ODBC的C/S专用共享库
gdlcs.dll	X	O	O	ODBC的C/S专用共享库
README	X	O	O	read me
sundbjnigc.dll	X	O	O	SUNDB应用开发库(JDBC D/A mode)

Table 2-11 lib目录（Windows client）

文件名	服务器	客户端	说明
README	O	O	read me
sql.h	O	O	ODBC头文件
sqlca.h	O	O	ODBC头文件
sqlext.h	O	O	ODBC头文件
sqltypes.h	O	O	ODBC头文件
sqlucode.h	O	O	ODBC头文件
sundb.h	O	O	SUNDB ODBC应用程序开发用头文件
sundbtypes.h	O	O	SUNDB ODBC数据类型明细文件
xa.h	O	O	标准XA头文件
sundbxa.h	O	O	SUNDB XA头文件
sundbesql.h	O	O	嵌入式SQL头文件

Table 2-12 include目录（Unix）

文件名	服务器	客户端	说明
README	X	O	read me
sundb.h	X	O	SUNDB ODBC应用程序开发用头文件
sundbtypes.h	X	O	SUNDB ODBC数据类型明细文件

文件名	服务器	客户端	说明
sundbxa.h	X	O	SUNDB XA头文件
sundbesql.h	X	O	嵌入式SQL头文件
sqlca.h	X	O	ODBC头文件

Table 2-13 include目录（Windows Client）

文件名	服务器	客户端	说明
README	O	X	read me

Table 2-14 license目录

文件名	服务器	客户端	说明
README	O	O	read me
sundb_error.msg	O	O	报错消息文件

Table 2-15 msg目录

文件名	说明
README	read me
sundb.property.conf	数据库运行参数文本文件
sundb.listener.conf	Listener参数文件
sundb.invited.conf	数据库接受连接的客户端管理文件

文件名	说明
sundb.excluded.conf	数据库拒绝连接的客户端管理文件
sundb.gagent.conf	gagent专用设置文件
tablediff.conf	tablediff设置文件
cyclone.master.conf	Cyclone master专用基本文件
cyclone.slave.conf	Cyclone slave专用基本文件
logmirror.master.conf	LogMirror master专用基本文件
logmirror.slave.conf	LogMirror slave专用基本文件
odbc.ini	odbc配置模板
gsql.ini	gsql配置模板
glogin.sql	启动gsql时执行的命令目录
sundb.glocator.conf	gLocator专用设置文件
cyfile.conf	cyfile专用设置文件
clustone.master.conf	clustone master专用基本文件
clustone.slave.conf	clustone slave专用基本文件

Table 2-16 conf目录

文件名	说明
README	read me

Table 2-17 db目录

文件名	说明
README	read me

Table 2-18 wal目录

文件名	说明
README	read me

Table 2-19 archive_log目录

文件名	说明
README	read me

Table 2-20 backup目录

文件名	说明
README	read me

Table 2-21 trc目录

文件名	说明
README	read me

Table 2-22 data目录

安装SUNDB软件

安装SUNDB之前先介绍操作系统以及环境设置

内核参数

共享内存（Shared Memory）

共享内存是IPC（Inter Process Communication）的一种指多个应用程序为了共享数据而使用的内存SUNDB的gsqlgloader以及基于ODBC的应用程序在CS (Client – Server) 连接时使用共享内存另外 由于用于运行的表空间均在共享内存中生成因此设置要比其他程序的推荐值更加精确

以下为用于安装SUNDB的共享内存所需的参数以及推荐值

参数名	说明	推荐值	备注
shmmax	单个共享内存段的最大大小	大于最大数据文件大小的值	设置为大于预想中的表空间中的最大数据文件大小的值
shmmni	系统可使用的最大共享内存段数	大于所有数据文件数量+1的值	设置为大于所有数据文件数量+1（用于SSA的共享内存段）的值

参数名	说明	推荐值	备注
shmall	所有共享内存段的总和（页数）	大于表空间结构总和的值	指系统可使用的共享内存的总页数一般情况下在使用大于8GB以上的共享内存时使用如果SUNDB的表空间总和为32G则shmall应大于此值

Table 2-23 共享内存相关内核参数

以下为表空间总和为32G时的shmall值的示例

```
kernel.shmmax = 34359738368
kernel.shmmni=4096
kernel.shmall = 8388609
```

- 假设shmmax值为32GPAGE_SIZE为4096 byte

```
8388609 = (34359738368 / 4096) + 1
```

- 此时shmall值应大于8388609

信号量（semaphore）

信号量（semaphore）与共享内存一样是IPC的一种是在操作系统控制多个使用资源的进程的技术通过信号量设置多个进程可以同时引用相关资源也可以在某一个进程正在使用该资源时使另一个进程等待该进程完成使用资源

SUNDB使用信号量来控制上述说明的对共享内存的访问顺序例如多个SUNDB客户端程序请求更

改相同数据时需要适当对其进行控制应根据SUNDB运行时产生的信号量操作设置适当的信号量参数值本文档中推荐Linux的常规值

以下为用于安装SUNDB的信号量推荐值

内核参数	说明	推荐值
semmsl	一个信号量组合中设置的信号量的数量	250
semmni	信号量组合的数量	128
semmns	信号量组合的总数量(semmni * semmsl)	32000
semopm	每个系统调用可执行的最大信号量数量	100

Table 2-24 信号量相关内核参数推荐值

RedhatUbuntu等基于Linux的系统中如果创建IPC资源的用户从systemd管理的会话列表中注销则会自动删除相应的IPC资源因此为了防止删除信号量系统应如下进行设置（内核3.0.0以上）

```
# cp -i /etc/systemd/logind.conf /etc/systemd/logind.conf_prev


# cat /etc/systemd/logind.conf
[Login]
#NAutoVTs=6
#ReserveVT=6
...
RemoveIPC=no
```

- 修改后执行

```
# systemctl restart systemd-logind
```

网络（Network）

backlog是TCP socket监听期间等待被接受的socket的队列长度SUNDB glsnr的backlog设置为glsnr的config文件的backlog如果系统中backlog的最大值大于somaxconn则将其设置为somaxconn并拓展somaxconn

SUNDB在C/S的共享模式下运行时所使用的UDS(Unix Domain Socket)的队列长度设置为max_dgram_qlen当客户端同时访问时如果该值过小则会导致glsnrgbalancer和gdispatcher之间产生通信瓶颈

以下为用于安装SUNDB使用的网络参数推荐值

内核参数	说明	推荐值
somaxconn	listen backlog的最大值	1024
max_dgram_qlen	Unix Domain Socket Queue大小	256

Table 2-25 网络相关内核参数推荐值

应用参数

一次性应用时执行如下操作（重启系统时需要重新应用）

```
[SHELL]> echo   34359738368   >  /proc/sys/kernel/shmmax
[SHELL]> echo   8388608 >   /proc/sys/kernel/shmall
```

```
[SHELL]> echo   4096   >  /proc/sys/kernel/shmmni

[SHELL]> echo   250 32000 100 128  /proc/sys/kernel/sem

[SHELL]> echo   1024   >  /proc/sys/net/core/somaxconn

[SHELL]> echo   256   >  /proc/sys/net/unix/max_dgram_qlen
```

如果要在系统重启时应用则在/etc/sysctl.conf中执行以下操作

```
# shared memory

kernel.shmmax = 34359738368

kernel.shmall = 8388608

kernel.shmmni = 4096


# semaphore

kernel.sem = 250 32000 100 128


# network

net.core.somaxconn = 1024

net.unix.max_dgram_qlen = 256
```

通过执行以下命令应用上述内容

```
[SHELL]> sysctl  -p
```

查看参数

通过以下命令可以查看设置的参数

```
[SHELL]> ipcs -l


------ Shared Memory Limits --------
max number of segments = 4096
max seg size (kbytes) = 33554432
max total shared memory (kbytes) = 33554432
min seg size (bytes) = 1


------ Semaphore Limits --------
max number of arrays = 128
max semaphores per array = 250
max semaphores system wide = 32000
max ops per semop call = 100
semaphore max value = 32767
```

解压

SUNDB的软件包以压缩形式提供通过解压来完成基本安装

以下为安装SUNDB软件包的简单示例

```
##  $SUNDB_HOME=/home/SUNDB/sundb-mercury.2.1.0-linux-x86_64/


[SHELL]> gzip –d sundb-mercury.2.1.0-linux-x86_64.tar.gz


[SHELL]> tar -xvf sundb-server-mercury.2.1.0-linux-x86_64.tar
```

```
sundb-server-mercury.2.1.0-linux-x86_64/sundb_home/include/sqlext.h
sundb-server-mercury.2.1.0-linux-x86_64/sundb_home/include/sundb.h
sundb-server-mercury.2.1.0-linux-x86_64/sundb_home/include/sqlca.h
…
```

完成解压后可根据所需更改创建为<package_file_name>的目录名而且要相应的更改$SUNDB_HOME与$SUNDB_DATA环境变量

有关解压后重建的目录的更多相关说明请参考**SUNDB安装包配置**

环境变量设置

解压SUNDB安装包后为了后续运行SUNDB软件并开发应用程序将$SUNDB_HOME目录下生成的bin与lib如下添加到PATH和LD_LIBRARY_PATH中（开发SUNDB客户端应用程序时用户应始终在编译选项的include文件目录中插入$SUNDB_HOME/include）

```
export PATH=$SUNDB_HOME/bin:$PATH
export LD_LIBRARY_PATH=$SUNDB_HOME/lib:$LD_LIBRARY_PATH
```

使用SUNDB需要环境变量由于在安装时也需要引用某些变量因此应在安装之前提前设置

环境变量	说明	备注
SUNDB_HOME	安装SUNDB二进制文件的目录路径	作为运行SUNDB时引用的变量需要提前将安装SUNDB的目录添加到环境变量

环境变量	说明	备注
SUNDB_DATA	创建SUNDB数据库实例的位置	创建或运行SUNDB数据库时引用的变量
PATH	SUNDB执行文件的目录路径	需要在在没有绝对路径的情况下执行SUNDB的各种二进制时设置
LANG	终端字符集	• 创建SUNDB数据库时如果与所描述的字符集不同则除了英文数字特殊符号外的其他字符可能显示不正确或string相关函数的操作会出错 • 需要设置对应GB18030SQL_ASCII, UHC, UTF8的locale • 例：export LANG=ko_KR.utf8

Table 2-26 安装SUNDB时引用的OS环境变量

```
##  $SUNDB_HOME=/home/SUNDB/sundb-mercury.2.1.0-linux-x86_64/


[SHELL]> gzip –d sundb-mercury.2.1.0-linux-x86_64.tar.gz


[SHELL]> tar -xvf sundb-server-mercury.2.1.0-linux-x86_64.tar
sundb-server-mercury.2.1.0-linux-x86_64/sundb_home/include/sqlext.h
sundb-server-mercury.2.1.0-linux-x86_64/sundb_home/include/sundb.h
sundb-server-mercury.2.1.0-linux-x86_64/sundb_home/include/sqlca.h
```

> ...

删除数据库

为了重新创建SUNDB数据库而需要删除现有数据库时删除数据文件控制文件重做日志文件和归档日志文件（使用归档日志时）即可

以下为删除SUNDB数据库的示例

```
## $SUNDB_BASE=/home/SUNDB/sundb-mercury.2.1.0-linux-x86_64/
## $SUNDB_DATA=/home/SUNDB/sundb-mercury.2.1.0-linux-x86_64/
                    sundb_data/
## $SUNDB_HOME=/home/SUNDB/sundb-mercury.2.1.0-linux-x86_64/
                    sundb_home/
[SHELL]> rm -rf $SUNDB_DATA/db/*.dbf
[SHELL]> rm -rf $SUNDB_DATA/wal/*.ctl
[SHELL]> rm -rf $SUNDB_DATA/wal/*.log
[SHELL]> rm -rf $SUNDB_DATA/archive_log/*.log
```

根据用户的设置数据文件或归档日志文件等的位置会有所不同

删除

SUNDB安装包以压缩文件的形式提供因此不需要特定的删除规则关闭数据库后删除已安装的目录即可

以下为删除SUNDB安装包的简单示例

```
## $SUNDB_BASE=/home/SUNDB/sundb-mercury.2.1.0-linux-x86_64/

## $SUNDB_DATA=/home/SUNDB/sundb-mercury.2.1.0-linux-x86_64/

                    sundb_data/

## $SUNDB_HOME=/home/SUNDB/sundb-mercury.2.1.0-linux-x86_64/

                    sundb_home/

[SHELL]> rm -rf $SUNDB_DATA

[SHELL]> rm -rf $SUNDB_HOME

[SHELL]> rm -rf $SUNDB_BASE
```

```
## ~/.bash_profile

$SUNDB_BASE=/home/SUNDB/sundb-mercury.2.1.0-linux-x86_64/ ❶ 删除

$SUNDB_DATA=/home/SUNDB/sundb-mercury.2.1.0-linux-x86_64/

                    sundb_data/ ❷ 删除

$SUNDB_HOME=/home/SUNDB/sundb-mercury.2.1.0-linux-x86_64/

                    sundb_home/ ❸ 删除
```

创建数据库

完成参数设置后创建数据库可使用$SUNDB_HOME/bin/gcreatedb命令超级数据库以下为gcreatedb命令的使用方法

```
[SHELL]> gcreatedb --help

Usage
```

```
    gcreatedb [options]


Options:


    --cluster        cluster system (if not specified, stand-alone system)
    --db_name        database name
    --db_comment     database comment
    --timezone       timezone ( {+/-}{TZH:TZM} )
    --character_set  character set
                       SQL_ASCII
                       UTF8
                       UHC
                       GB18030
    --char_length_units  char length units
                           OCTETS
                           CHARACTERS
    --member         local cluster member name
    --host           cluster ip address or host name
    --port           cluster port number
    --silent         suppresses the display of the result message
    --help           print help message


examples:
```

```
    gcreatedb --db_name="sundb" --db_comment="sundb database" --
timezone="+09:00" --character_set="UTF8" --char_length_units="OCTETS" --
silent
```

创建数据库时将引用$SUNDB_HOME/conf/sundb.properties.conf以各个***_TABLESPACE_SIZE值在sundb.properties.conf的‘SYSTEM_TABLESPACE_DIR’路径下创建表空间文件

创建要参与集群系统的数据库时必须指定--cluster选项后执行

以下为gcreatedb命令的执行参数相关说明

执行参数	说明
--cluster	用于集群系统的数据库 如果省略则创建单机版数据库
--db_name	数据库名称 如果省略则设置为'sundb'
--db_comment	数据库相关说明 如果省略则设置为'sundb database'
--timezone	时区 如果省略则设置为TIMEZONE参数

执行参数	说明
--character_set	数据库字符集 SUNDB支持以下四种字符集 如果省略则设置为CHARACTER_SET参数 • GB18030：中文简体 • SQL_ASCII：仅支持ASCII字符的字符集 • UHC：统一韩文代码（Unified Hangul Code） • UTF8：（Unicode 转换格式 – 8 ）
--char_length_units	字符长度的单位 如果省略则设置为CHAR_LENGTH_UNITS参数 • OCTETS：1 个字节识别为1个字符 • CHARACTERS：1个字符（n个字节）识别为1个字符
--member	要在集群系统中使用的本地数据库的成员名称 如果省略则使用LOCAL_CLUSTER_MEMBER参数中设置的值
--host	用于集群系统成员之间通信的本地成员的主机名称或者IP地址使用主机名称时使用系统的第一个IPv4地址 如果省略则使用LOCAL_CLUSTER_MEMBER_HOST参数中设置的值
--port	用于集群系统成员之间通信的本地成员的TCP监听端口 如果省略则使用LOCAL_CLUSTER_MEMBER_PORT参数中设置的值
--silent	隐藏显示信息

执行参数	说明
--help	显示帮助

Table 2-27 gcreatedb参数

创建数据库时需要考虑以下事项

- 内核参数共享内存的设置
  - 如果上述说明的共享内存设置中指定的大小小于$SUNDB_HOME/conf/sundb.properties.conf中描述的大小则无法创建数据库
- 表空间大小
  - 创建数据库时创建的表空间文件在启动SUNDB时分配并使用与表空间相同大小的内存因此数据表空间中即使没有实际的用户数据也会占用内存空间因此在编写sundb.properties.conf时除了共享内存外还需要考虑SUNDB启动器上的可用内存后创建数据库
- 是否使用集群系统
  - 创建数据库时应指定该数据库是否作为集群系统的成员参与集群系统即作为单机版使用时执行gcreatedb时应省略--cluster选项但使用为集群系统时必须指定该选项

成功创建数据库后可在sundb.properties.conf的‘SYSTEM_TABLESPACE_DIR’中描述的路径下查看如下表空间文件

```
[SHELL]> gcreatedb
Database created

[SHELL]> gcreatedb --db_name="TEST_DB"                    \
```

```
                    --db_commnet="test database comment"  \
                    --timezone="+09:00"                   \
                    --character_set="UHC"                 \
                    --char_length_units="OCTETS"
Database created

[SHELL]> ls $SUNDB_DATA/db
system_data.dbf  system_dict.dbf  system_undo.dbf
```

以下为创建集群系统中使用的数据库的示例

```
[SHELL]> gcreatedb --cluster
Database created

[SHELL]> gcreatedb --cluster --member=G1N1
Database created

[SHELL]> gcreatedb --cluster --member=G1N1 --host=127.0.0.1 --port 10101
Database created

[SHELL]> gcreatedb --cluster                          \
                    --db_name="TEST_DB"               \
                    --home=$SUNDB_DATA           \
                    --host=127.0.0.1                  \
                    --port=10101                      \
                    --db_comment="g1n1 db comment"    \
```

```
                    --timezone="+09:00"                \
                    --character_set="UHC"              \
                    --char_length_units="OCTETS"
Database created


[SHELL]> ls $SUNDB_DATA/db
system_data.dbf  system_dict.dbf  system_trans.dbf system_undo.dbf
```

构建字典模式信息

创建如下模式以查询系统及对象信息

> Caution:
>
> 创建数据库后必须创建如下模式否则获取对象结构信息的ODBCJDBC的Catalog API（例如SQLTables（）函数等）会出现误操作也无法与第三方工具联动

- DICTIONARY_SCHEMA：由查询DBA_*, ALL_*, USER_*等对象信息的视图和表组成
- INFORMATION_SCHEMA：由包含在SQL标准的INFORMATION_SCHEMA中的视图和表组成
- PERFORMANCE_VIEW_SCHEMA：由组合Fixed table的信息来查询系统信息的视图组成

通过该模式中包含的视图和表可以方便地查询系统信息

SUNDB实例启动到OPEN状态后按照如下方式执行sql文件且需要在创建数据库后最初执行一次

> Note:

> 构建字典模式信息的脚本分为单机版用与集群系统用因此必须使用对应脚本来构建信息

以下为使用单机版数据库专用脚本构建字典模式信息的方法

```
% gsql --as sysdba --import
$SUNDB_HOME/admin/standalone/DictionarySchema.sql
% gsql --as sysdba --import
$SUNDB_HOME/admin/standalone/InformationSchema.sql
% gsql --as sysdba --import
$SUNDB_HOME/admin/standalone/PerformanceViewSchema.sql
```

以下为使用集群系统专用脚本构建字典模式信息的方法

```
% gsql --as sysdba --import $SUNDB_HOME/admin/cluster/DictionarySchema.sql
% gsql --as sysdba --import
$SUNDB_HOME/admin/cluster/InformationSchema.sql
% gsql --as sysdba --import
$SUNDB_HOME/admin/cluster/PerformanceViewSchema.sql
```

2.3 数据库存储结构管理

本章介绍组成SUNDB实例的数据库组件

数据库存储结构

SUNDB数据库主要分为内存区域和磁盘区域内存区域是由多个共享内存块组成的表空间集合不会被Replace操作写入到磁盘上

磁盘区域包含每个表空间的共享内存中存在一个并保存数据文件和实例配置信息的控制文件存储实例环境配置的属性文件以及用于恢复数据库的在线重做日志文件

控制文件

控制文件是在数据库磁盘记录物理存储信息启动实例时最先被读取并了解数据库状态控制文件记录如下内容

- 上一个检查点的实例的状态
- 上次启动时的事务持久性模式(Transaction Durability Mode)(CDS/TDS)
- 在线重做日志文件信息
- 各表空间的数据文件信息

在线重做日志文件(Online redo log file)

数据库非正常结束时会出现未来得及写入数据文件的变更信息为了重启实例时恢复这些变更信息在线重做日志文件中存储事务在该实例执行的所有变更数据库的操作信息默认情况下在创建

数据库时会以LOG_FILE_SIZE参数中指定的大小生成四个在线重做日志文件需要时也可以增加这些重做日志文件以循环方式进行再使用

在线重做日志文件切换到下一个文件时会产生检查点(checkpoint)部分脏页将回写到对应的数据文件如果检查点操作延迟导致下一个在线重做日志文件中的变更页尚未回写到磁盘时（ACTIVE状态时）则所有事务会暂停直到完成检查点因此根据应用程序的性质创建大小合适的在线重做日志文件将有助于提高数据库系统性能

回滚段(Undo segment)

回滚段是为了回滚部分或全部事务而记录变更操作之前的镜像的Segment每个执行变更操作的事务将分配一个回滚段回滚段存储于MEM_UNDO_TBS表空间并以防产生多个变更事务或通过一个事务执行大量的变更操作（批量删除等）建议确保足够的Undo表空间

数据文件(Data File)

数据文件存储表空间中存储的表/索引的内容

数据文件构成如下

- Page
  - 是数据库 I/O的最小单位目前定义为8K字节
- Extent
  - 一定数量的连续的Page集合Segment从表空间中分配空间的最小单位
  - 根据各个表空间Extent的大小会有所不同
- Segment
  - 特定类型数据的集合对象由Extent的集合构成

特殊情况下类似MEM_TEMP_TBS表空间等临时表空间不需要执行重做日志也不创建数据文件

表空间

数据库分为包含表与索引等的逻辑结构的表空间SUNDB的表空间分为内存表空间和磁盘表空间按照表空间的各个数据文件创建共享内存并在访问page时不产生额外的磁盘I/O的是内存表空间使用系统的buffer cache访问page的是磁盘表空间查询V$TABLESPACE表时输出存在于当前数据库的表空间信息

SUNDB默认支持以下表空间

所有者	名称	说明
SYSTEM	DICTIONARY_TBS	存储运行数据库所需的默认字典表
	MEM_UNDO_TBS	存储回滚段和事务信息
	MEM_DATA_TBS	如用户生成模式对象时未指定则默认存储于此表空间中
	DISK_DATA_TBS	用户默认可使用的磁盘表空间
	MEM_TEMP_TBS	查询中生成的未指定临时表或表空间的索引生成在该位置由于不发生日志记录重启实例时重建索引
	MEM_TRANS_TBS	用于恢复集群系统中的全局事务仅在配置成集群系统时生成
USER	用户自定义	用户可生成表空间用于整合管理特定用途的表

Table 2-28 SUNDB表空间种类

表空间类型

表空间具有以下5种类型

- DICT
  - 用于存储字典表与索引的表空间类型
- DATA
  - 用于存储表与索引等常规模式对象的表空间类型
  - 是重做日志记录与page flush的对象
- UNDO
  - 用于存储回滚段的表空间类型
  - 是重做日志记录与page flush的对象
- TEMPORARY
  - 用于存储在SELECT查询中创建的临时表和无日志记录索引(no-logging index)的表空间类型
  - 不发生重做日志记录与page flush
- TRANSACTION (仅限集群)
  - 用于在集群系统中恢复全局事务的表空间类型

查看数据库存储结构信息

本章介绍可查看上述多个数据库存储结构相关信息的方法

控制文件信息(Control file)

控制文件以二进制形式存储因此查看其内容需使用如下"gdump"工具

```
[SHELL]> gdump CONTROL control_0.ctl
```

在线重做日志文件信息(Online redo log file)

使用gdump工具查询控制文件将显示各在线重做日志文件的名称与当前状态

数据文件信息

使用gdump工具查询控制文件将显示各表空间中的数据文件及其状态另外查询V$DATAFILE表可显示当前状态

表空间信息

使用gdump工具查询控制文件将显示当前数据库中的各表空间的名称与状态也可通过SQL查询V$TABLESPACE表

参数信息

可打开文本文件 $SUNDB_Data/conf/sundb.properties.conf 文件查看在线状态下也能通过查询V$PROPERTYV$DB_PROPERTY表来查看当前应用的参数值

常用的数据存储操作

可对存储数据的表空间进行如下操作

创建表空间

可使用如下命令创建内存磁盘USER DATA表空间

```
gSQL> CREATE TABLESPACE TEST_TBS DATAFILE 'TEST_TBS.dbf' SIZE 10M;

Tablespace created.

gSQL> CREATE MEMORY TABLESPACE TEST_TBS DATAFILE 'TEST_MEM_TBS.dbf' SIZE
10M;

Tablespace created.

gSQL> CREATE DISK TABLESPACE TEST_TBS DATAFILE 'TEST_DISK_TBS.dbf' SIZE
10M;

Tablespace altered.

gSQL> CREATE DISK TABLESPACE DISK_TBS DATAFILE 'TEST_DISK_TBS.dbf' SIZE
10M AUTOEXTEND OFF;

Tablespace altered.
```

临时表空间没有数据文件因此通过以下语句创建

```
gSQL> CREATE TEMPORARY TABLESPACE TEST_TEMP_TBS MEMORY 'TEST_TEMP_TBS'
SIZE 10M;

Tablespace created.
```

查询表空间使用情况

表空间的空间由多个连续Page组成的Extent为单位进行分配或释放可如下查询特定表空间的一个Extent大小（BYTE）

```
gSQL> SELECT EXTENT_SIZE FROM V$TABLESPACE WHERE TBS_NAME = 'TEST_TBS';


EXTENT_SIZE
-----------
     262144


1 row selected.
```

可以使用D$TABLESPACE_EXTENT表查询表空间中所有Extent的状态使用中的Extent的STATE列显示为“U” free状态的Extent显示为“F”因此当前表空间中的剩余空间的大小（Extent数量）可如下进行计算

```
gSQL> SELECT COUNT(*) FROM D$TABLESPACE_EXTENT('TEST_TBS') WHERE STATE =
'F';


COUNT(*)
--------
      38


1 row selected.
```

根据以上结果TEST_TBS的当前剩余空间为38 * 262144 = 9437184 Byte

更改表空间

更改表空间的方法有“添加/删除数据文件（Add/Remove Data File)”（临时表空间的情况是内存）与“在线/离线（Online/Offline）”

添加/删除数据文件（或内存）

运行数据库的过程中由于表空间的空间不足时可通过以下语句追加分配数据表空间

```
gSQL> ALTER TABLESPACE TEST_TBS ADD DATAFILE 'TEST_TBS2.dbf' SIZE 10M;


Tablespace altered.
```

临时表空间可如下添加空间与数据表空间不同赋予内存的名称该名称必须是数据库中唯一的名称

```
gSQL> ALTER TABLESPACE TEST_TEMP_TBS ADD MEMORY 'TEST_TEMP_TBS2' SIZE 10M;


Tablespace altered.
```

可通过以下语句删除数据表空间但如果使用了部分空间则无法删除

```
gSQL> ALTER TABLESPACE TEST_TBS DROP DATAFILE 'TEST_TBS2.dbf';


Tablespace altered.
```

同样可通过以下语句回收临时表空间

```
gSQL> ALTER TABLESPACE TEST_TEMP_TBS DROP MEMORY 'TEST_TEMP_TBS2';


Tablespace altered.
```

离线模式下的表空间

如果要在表空间中执行移动数据文件位置等操作需要将该表空间更改为离线模式可通过如下语句更改为离线模式

```
gSQL> ALTER TABLESPACE TEST_TBS OFFLINE;


Tablespace altered.
```

用户可通过如下操作将表空间切换至在线模式

```
gSQL> ALTER TABLESPACE TEST_TBS ONLINE;


Tablespace altered.
```

更改磁盘表空间的数据文件自动扩张属性

磁盘表空间的数据文件可以按照初始大小创建后根据需要自动扩张至最大大小自动扩张属性可使用如下语句进行更改（on/off）更改自动扩张属性时也可同时更改数据文件的自动扩张大小和最大大小

```
gSQL> ALTER DATABASE DATAFILE 'TEST_DISK_TBS.dbf' AUTOEXTEND ON;

Database altered.

gSQL> ALTER DATABASE DATAFILE 'TEST_DISK_TBS.dbf' AUTOEXTEND OFF;

Database altered.

gSQL> ALTER DATABASE DATAFILE 'TEST_DISK_TBS.dbf' AUTOEXTEND ON NEXT 10M
MAXSIZE 20M;

Database altered.
```

重命名

可通过以下语句更改表空间名称

```
gSQL> ALTER TABLESPACE TEST_TBS RENAME TO TEST_TBS2;

Tablespace altered.
```

如果要更改表空间中数据文件的位置首先要将表空间切换到离线模式之后使用OS命令移动该数据文件再通过ALTER TABLESPACE语句重命名表空间的数据文件最后将表空间切换为在线模式

```
gSQL> ALTER TABLESPACE TEST_TBS OFFLINE;
Tablespace altered.
```

```
gSQL> ALTER TABLESPACE TEST_TBS RENAME DATAFILE 'TEST_TBS.dbf' TO
'TEST_TBS_1.dbf';

ERR-42000(16164): file does not exist :
ALTER TABLESPACE TEST_TBS RENAME DATAFILE 'TEST_TBS.dbf' TO
'TEST_TBS_1.dbf'
                                                                    *
ERROR at line 1:
```

- 使用OS命令更改文件名

```
gSQL> ALTER TABLESPACE TEST_TBS RENAME DATAFILE 'TEST_TBS.dbf' TO
'TEST_TBS_1.dbf';

Tablespace altered.

gSQL> ALTER TABLESPACE TEST_TBS ONLINE;

Tablespace altered.
```

删除表空间

可如下删除不需要的表空间INCLUDING后面的语句是可选的如果使用该选项将删除该表空间中的所有内容（模式对象）与数据文件

```
gSQL> DROP TABLESPACE TEST_TBS INCLUDING CONTENTS AND DATAFILES;
```

```
Tablespace dropped.
```

存储模式

为了在特定环境下使其性能最大化SUNDB以实例为单位设置存储模式并使用存储模式是为了提高事务的操作性能指定事务放弃ACID属性中哪一个部分

SUNDB目前支持以下两种类型的存储模式

- TDS (事务数据存储) 模式
  - TDS模式为SUNDB 数据库的默认存储模式
  - 在此实例执行的所有事务均为记录回滚日志和重做日志的通用的DBMS存储模式因此可回滚事务由于定期执行Checkpoiont因此在数据库异常结束时也能恢复数据文件与在线重做日志文件
- CDS（并发数据存储）模式
  - 在此实例中执行的所有事务只记录回滚日志不记录重做日志因此事务可以处理所有运行时的错误且可以回滚但在实例异常结束或使用Shutdown abort命令结束时由于不发生Checkpoint因此会丢失执行的所有变更信息（不支持恢复功能）CDS模式支持事务之间的并发性控制因此多个事务同时访问同一个对象时也能确保事务正常操作
  - CDS模式主要用于Cache服务器等频繁执行查询及更新操作但不需要保障持久性的以运行信息为主的数据库

实例启动时通过参数设置来启动存储模式事务不能在不同存储模式下运行而且不能在实例的在线模式下更改存储模式因此需要慎重考虑后决定模式

2.4 模式对象(Schema Object)管理

Schema Object

模式对象是由用户创建的逻辑结构SUNDB支持table, index, synonym, view, sequence, constraint, storedProcedure等模式对象

模式对象管理权限

目前SUNDB支持用户和权限 因此并非所有用户都能共享所有创建的对象需要赋予权限才可以共享

表管理

本章介绍表的概要与查看表相关信息的方法以及创建表/变更表/加载数据/删除数据的方法

表

表是存储用户数据的最基本的存储单位由列（Column）和行（Row）组成

表类型

SUNDB目前不支持聚簇表(clustered table)分区表(partitioned table)仅支持普通的heap table（数据的存储顺序与特定列的排序顺序无关）

- 仅支持基本的Heap Table
- 支持主键/唯一/非空约束
- 支持的数据类型
  - BOOLEAN
  - SMALLINT, INTEGER, BIGINT, REAL, DOUBLE, NUERMIC, FLOAT
  - CHAR(MAX 2000), VARCHAR(MAX 4000), BINARY, VARBINARY, LONG VARCHAR, LONG VARBINARY
  - DATE, TIME, TIMESTAMP, INTERVAL (支持TIME/TIMESTAMP的WITH/WITHOUT TIMEZONE)
  - 不支持BLOB类型
- 列数量 / 索引数量 / 约束条件数量无限制
- 通过"SELECT * FROM TABLES;"查询可查看全部表信息

如果在特定表里存储了大量行导致表的大小过大时即使执行批量删除也不会把空间重新释放给表空间但是可通过TRUNCATE操作把所有原有空间返回给表空间

将SUNDB以集群系统的结构使用时可以根据用户所需的分配策略将表数据分散到多个节点并可进行存储与查询详细信息请参考**表管理**

索引管理

本节介绍索引概要以及创建和删除索引的方法

索引概要

索引是连接到表的附加模式对象使用索引可以提高查找特定条件对应的行的位置的性能如果要

获取的列为相应索引的key列时也可获取该行的列值

SUNDB可以为表创建所需数量的索引但是过多的索引会增加表的插入/更改/删除操作的负荷可能降低数据库性能

表的主键或唯一约束会自动在相应列上创建索引

索引属性

- 支持B-Link Tree形式的索引
  - SUNDB默认提供以B-Link Tree方式实现的索引
- 索引的单个节点大小为8 Kbyte
- 最大关键列（key column）数量为32个最大key长度为2000字节
- 支持唯一索引
- 支持升序/降序NULLS FIRSTNULLS LAST
  - 可定义是按升序（ASC）还是按降序（DESC）顺序对索引的关键列进行排序
  - 可定义将索引的关键列中的NULL值放在最前面（NULLS FIRST）还是放在最后面（NULLS LAST）
- 可通过"SELECT * FROM INDEXES;"查询查看全部索引信息
- 索引也与表相同由Segment实现因此可以通过与表类似的方法计算索引的大小

序列

序列是一个用户可生成唯一编号的模式对象

可如下创建

```
gSQL> CREATE SEQUENCE customers_seq START WITH 1000 INCREMENT BY 1 NOCACHE
NOCYCLE;


Sequence created.
```

可利用NEXTVAL使用序列

```
gSQL> SELECT customers_seq.NEXTVAL FROM dual;
```

可如下删除序列

```
gSQL> DROP SEQUENCE customers_seq;
```

当由集群系统构成SUNDB并使用时用户创建序列时将自动创建全局序列对象此全局序列对象管理集群系统中的所有成员节点共享使用的序列值的全局池各个成员节点调用NEXTVAL时从全局对象分配指定数量的序列值并使用详细内容请参考**全局序列**

2.5 用户管理

创建用户

只有SYS用户与拥有CREATE USER ON DATABASE权限的用户才能创建SUNDB数据库的用户而且为了连接创建的用户需要拥有CREATE SESSION ON DATABASE权限

语句如下

```
<user definition> ::=
    CREATE USER user_identifier IDENTIFIED BY password
    [ DEFAULT TABLESPACE tablespace_name ]
    [ TEMPORARY TABLESPACE tablespace_name ]
    [ INDEX TABLESPACE tablespace_name ]
```

语句规则及参数如下

- user_identifier
  - 要创建的用户的名称
  - 不能存在相同的用户名或角色名
  - user_identifier的长度必须小于128 byte
- password
  - 要创建的用户密码以加密形式存储
  - 密码的长度必须小于128 byte
  - 密码区分大小写

- DEFAULT TABLESPACE tablespace_name
  - 指定用户要创建的表索引（LOGGING）等对象存储的默认表空间
  - 省略DEFAULT TABLESPACE子句时将指定为创建数据库时（<database definition>）定义的默认数据存储表空间(MEM_DATA_TBS)
- TEMPORARY TABLESPACE tablespace_name
  - 指定用户要创建的临时表索引（NO LOGGING）存储查询处理过程中的创建的中间结果的表空间
  - 省略TEMPORARY TABLESPACE子句时将指定创建数据库时（<database definition>）定义的默认临时表空间 (MEM_TEMP_TBS)
- INDEX TABLESPACE tablespace_name
  - 指定用户要创建的存储索引对象的默认表空间
  - 省略INDEX TABLESPACE子句时默认值为 INDEX TABLESPACE NULL

删除用户

删除SUNDB数据库已创建的用户用户范围及访问权限与创建用户相同

语句如下

```
<drop user statement> ::=
    DROP USER [ IF EXISTS ] user_identifier [ <drop behavior> ]
    ;

<drop behavior> ::=
      RESTRICT
```

```
    | CASCADE
```

语句规则及参数如下

- IF EXISTS
  - 用户不存在时不报错
- user_identifier
  - 要删除的用户的名称
  - 无法删除数据库创建时自动创建的“SYS”等用户
  - 不删除表空间对象等由user_identifier创建但非所有者的对象
- drop behavior
  - 省略时默认值为RESTRICT

更改用户

更改在SUNDB数据库中创建的用户的定义普通用户需要拥有ALTER USER权限如果是user_identifier用户可在没有权限的情况下执行

语句如下

```
<alter user statement> ::=

      ALTER USER user_identifier <alter user action>

    | ALTER USER PUBLIC <alter schema path>

    ;


<alter user action> ::=
```

```
      <alter password>
    | <alter profile>
    | <alter default tablespace>
    | <alter temporary tablespace>
    | <alter index tablespace>
    | <alter schema path>

<alter password> ::=
    IDENTIFIED BY new_password [ REPLACE old_password ]

<alter profile> ::=
    PROFILE { profile_name | DEFAULT | NULL }

<password expire> ::=
    PASSWORD EXPIRE

<account lock> ::=
    ACCOUNT { LOCK | UNLOCK }

<alter default tablespace> ::=
    DEFAULT TABLESPACE tablespace_name

<alter temporary tablespace> ::=
    TEMPORARY TABLESPACE tablespace_name
```

```
<alter index tablespace> ::=

    INDEX TABLESPACE { tablespace_name | NULL }


<alter schema path> ::=

    SCHEMA PATH ( { schema_name | CURRENT PATH } [, ...] )
```

语句规则及参数如下

- user_identifier
  - 要更改的用户名称
- <alter password>
  - 变更用户密码
  - IDENTIFIED BY new_password
    - 以加密方式存储新密码
    - 密码长度必须小于128 byte
    - 密码区分大小写
  - REPLACE old_password
    - 拥有ALTER USER ON DATABASE权限时可省略
    - 没有ALTER USER ON DATABASE权限时不可省略
    - 用户与user_identifier必须相同
- <alter profile>
  - 更改用于密码管理策略的profile
  - PROFILE profile_name
    - 分配用户创建的profile_name
  - PROFILE DEFAULT
    - 分配默认profile ”DEFAULT“

- PROFILE NULL
  - 不分配Profile

- <password expire>
  - 使用户的密码过期

- <account lock>
  - ACCOUNT LOCK
    - 锁定用户账号
  - ACCOUNT UNLOCK
    - 解除账号锁定

- <alter default tablespace>
  - 更改用户的默认表空间
  - tablespace_name应为数据表空间

- <alter temporary tablespace>
  - 更改用户的临时表空间
  - tablespace_name应为临时表空间

- <alter index tablespace>
  - 变更用户的索引表空间
  - 指定INDEX TABLESPACE tablespace_name
    - 指定数据表空间时为logging索引
    - 指定临时表空间时为no-logging索引
  - INDEX TABLESPACE NULL
    - 不指定索引表空间

2.6 SUNDB参数

SUNDB参数分为创建数据库时应用的参数与可在线/离线变更的参数只能在启动阶段的MOUNT阶段变更表空间和重做日志文件的存储路径

创建数据库时应用的参数

名称	说明
SYSTEM_MEMORY_DICT_TABLESPACE_SIZE	字典表空间的初始大小
SYSTEM_MEMORY_DATA_TABLESPACE_SIZE	系统数据表空间的初始大小
SYSTEM_MEMORY_UNDO_TABLESPACE_SIZE	系统回滚表空间的初始大小
SYSTEM_MEMORY_TEMP_TABLESPACE_SIZE	系统临时表空间的初始大小
LOG_BLOCK_SIZE	重做日志文件的块大小
LOG_FILE_SIZE	重做日志文件的初始大小
LOG_GROUP_COUNT	重做日志文件的数量
CHARACTER_SET	字符集
TIMEZONE	时区
CHAR_LENGTH_UNITS	字符长度单位

Table 2-29 创建数据库时应用的参数

启动数据库时应用的参数

SUNDB数据库有100个以上的参数以下为经常使用的部分参数

名称	说明
SHARED_MEMORY_STATIC_KEY	创建共享内存的关键值
SHARED_MEMORY_STATIC_SIZE	所创建的共享内存的大小
DATA_STORE_MODE	SUNDB实例的存储模式
LOG_BUFFER_SIZE	日志缓冲区的大小
LOG_DIR	重做日志的目录路径
PRIVATE_STATIC_AREA_SIZE	各会话的静态区域大小
CLIENT_MAX_COUNT	可访问会话的最大数量
PROCESS_MAX_COUNT	最大进程数量
NET_BUFFER_SIZE	各会话的通信缓冲区的大小

Table 2-30　启动数据库时应用的参数

2.7 SUNDB实用程序

gcreatedb

gcreatedb实用程序初始化SUNDB数据库并准备对外提供服务gcreatedb根据给予的参数在"SUNDB_DATA"环境变量的位置生成数据文件日志文件等语句如下

```
[SHELL]> gcreatedb --help
Usage

    gcreatedb [options]

Options:

    --cluster                cluster system (if not specified, stand-
alone system)
    --db_name                database name
    --db_comment             database comment
    --timezone               timezone ( {+/-}{TZH:TZM} )
    --character_set          character set
                               SQL_ASCII
                               UTF8
                               UHC
                               GB18030
```

```
    --char_length_units       char length units
                                OCTETS
                                CHARACTERS
    --member                  local cluster member name
    --host                    cluster ip address or host name
    --port                    cluster port number
    --silent                  suppresses the display of the result message
    --help                    print help message

examples:

    gcreatedb --db_name="sundb" --db_comment="sundb database" --
timezone="+09:00" --character_set="UTF8" --char_length_units="OCTETS" --
silent
```

创建数据库时引用$SUNDB_HOME/conf/sundb.properties.conf以***_TABLESPACE_SIZE值在sundb.properties.conf 的‘SYSTEM_TABLESPACE_DIR’路径中创建表空间文件

以下为gcreatedb命令的执行参数说明

执行参数	说明
--cluster	要在集群系统中使用的数据库 如果省略则创建单机版数据库
--db_name	数据库名称 如果省略则设置为'sundb'

执行参数	说明
--db_comment	数据库说明 如果省略则设置为'sundb database'
--timezone	时区 如果省略则设置为TIMEZONE参数
--character_set	数据库字符集 SUNDB支持以下四种字符集 如果省略则设置为CHARACTER_SET参数 • GB18030：中文简体 • SQL_ASCII：仅支持ASCII的字符集 • UHC：统一韩文代码 (Unified Hangul code) • UTF8 : Unicode转换格式 – 8
--char_length_units	字符长度单位 如果省略则设置为CHAR_LENGTH_UNITS参数 • OCTETS：将1 字节识别为1个字符 • CHARACTERS : 1个字符（n byte）识别为1个字符
--home	数据库主目录 查找属性文件并参照为各种DB文件的创建及存储位置 如果省略则使用SUNDB_DATA环境变量中设置的值

执行参数	说明
--member	要在集群系统中使用的本地数据库的成员名称 如果省略则使用LOCAL_CLUSTER_MEMBER参数中设置的值
--host	用于集群系统成员之间通信的本地成员的主机名称或者IP地址使用主机名称时使用系统的第一个IPv4地址 如果省略则使用LOCAL_CLUSTER_MEMBER_HOST参数中设置的值
--port	用于集群系统成员之间进行通信的本地成员的TCP监听端口 如果省略则使用LOCAL_CLUSTER_MEMBER_PORT参数中设置的值
--silent	隐藏输出信息
--help	显示帮助信息

Table 2-31 gcreatedb参数

gsql (SUNDB Interactive SQL Tool)

gsql是为管理SUNDB执行默认SQL的Interactive command line实用程序DBA可以通过gsql进行创建初始表模式或检查当前状态等操作

gsql执行方法如下

```
[SHELL]> gsql <userid> <passwd>


gSQL> CREATE TABLE T1 ( COL1 INTEGER );
```

```
create success

gSQL> \q

[SHELL]>
```

gloader (SUNDB数据上传/下载工具)

gloader是将数据库里的数据以文本格式下载为文件或将现有存储在文本文档中的数据上传到新的数据库的实用程序 gloader使用的文本数据文件格式为CSV（Comma-Separated Value）

gloader的使用方法如下

```
gloader userid passwd [export|import] control='control_file_name' \
data='data_file_name' log='log_file_name' bad='bad_file_name'
```

执行参数	说明
export | import	声明要通过gloader将现有表内容下载到data_file_name还是将当前存在于data_file_name的数据上传到control_file_name中指定的表
userid	指定要使用的userid
passwd	指定userid用户的密码
control	指定记录export或import操作的详细设置的文件路径

执行参数	说明
data	指定要export的目标数据文件或要import的数据文件
log	指定记录import或export的执行情况及经过时间等的日志文件路径
bad	指定记录import时因各种错误导致insert失败的数据记录的bad文件路径

Table 2-32 gloader参数

以下为控制文件内容的示例

```
TABLE  TEST_TBL
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
```

3.Cluster指南

3.1 SUNDB集群系统管理

本章介绍配置SUNDB数据库集群及管理集群系统的基础知识本章主要与单机版数据库进行对比介绍集群系统增加的部分或集群系统独有的特征在此之前需提前熟悉单机版数据库的教程

概述

如前一章所述SUNDB可以通过配置单机版数据库来使用还可以将多个数据库绑定到单个集群（cluster）中然后选择合适的数据分配策略使用即用户使用SUNDB集群系统时可以将海量数据的处理自定义分散到多个服务器上由此实现服务的高可用性及通过并行处理改善吞吐量

SUNDB集群系统由一个以上的集群组构成一个集群组由一个以上的集群成员组成不需要额外的应用程序服务器或元服务器并且应用程序通过连接到与数据服务器对应的集群成员来运行属于相同集群组的集群成员保持相同的数据副本（Replica）

创建各节点的数据库时需要提前确定使用单机版还是集群系统的SUNDB数据库如果要使用集群系统用户应当在各个节点创建数据库时添加集群相关选项

参数设置

与使用单机版数据库时相同用于构建集群系统的各种参数记述在每台服务器的$ SUNDB_DATA / conf / sundb.property.conf文件中即使使用集群系统各个数据库的TBS（表空间）LOGCONTROL FILE等主要相关参数的设置方式也与单机版相同但如果在同一服务器上创建多个数据库并将其配置为集群系统时要注意集群成员之间不应存在相同的文件路径及端口

以下是配置集群系统时需要考虑的主要参数及说明

参数	说明	默认值
SYSTEM_TABLESPACE_DIR	存储系统TBS的目录路径该目录中安装以下TBS • DICTIONARY_TBS • MEM_DATA_TBS • MEM_UNDO_TBS • MEM_TEMP_TBS • MEM_TRANS_TBS	'<SUNDB_DATA>/db'
SYSTEM_MEMORY_DICT_TABLESPACE_SIZE	字典表空间大小	256M
SYSTEM_MEMORY_DATA_TABLESPACE_SIZE	数据表空间大小	200M
SYSTEM_MEMORY_UNDO_TABLESPACE_SIZE	撤消表空间大小	32M
LOG_DIR	默认日志目录路径	'<SUNDB_DATA>/wal'
SYSTEM_LOGGER_DIR	系统日志目录路径	'<SUNDB_DATA>/trc'
CONTROL_FILE_COUNT	控制文件数量	2
CONTROL_FILE_0	第一个控制文件路径	'<SUNDB_DATA>/wal/control_0.ctl'
CONTROL_FILE_1	第二个控制文件路径	'<SUNDB_DATA>/wal/control_1.ctl'

参数	说明	默认值
LOCAL_CLUSTER_MEMBER	要在集群系统中使用的本地服务器的集群成员名称	‘G1N1’
LOCAL_CLUSTER_MEMBER_HOST	本地服务器的主机名	'127.0.0.1'
LOCAL_CLUSTER_MEMBER_PORT	本地服务器在集群系统中用于通信的TCP Listen端口	10101

Table 3-1 主要参数

集群成员名称与“主机 — 端口组合”在集群系统中必须是唯一的要想省略上述参数设置则在各个节点使用gcreatedb创建数据库时使用“--member”“--host”“--port”选项提供该信息通过参数或gcreatedb选项提供的成员信息在$ SUNDB_DATA / wal / location.ctl文件中存储并管理

要变更参数时变更文本参数文件（$ SUNDB_DATA / conf / sundb.properties.conf）或在环境变量中重新定义SUNDB_ <property_name>形式的变量即可参数文件中设置的值优先于环境变量中设置的值

后台进程

SUNDB集群系统具有在各个成员节点上用于管理实例的后台进程（gmaster）各个节点的gmaster内部由多个系统线程组成其中大多数与单机版系统相同但为了管理集群系统添加了以下系统线程

仅在启动以集群模式创建的数据库时才会调用以下线程：

- 集群恢复线程（Cluster Recover Thread）： 在集群系统中负责处理恢复全局事务
- 故障转移线程（Failover Thread）： 在集群系统中负责特定节点或网络故障时为故障成员重新选择脱机和协调器等的故障转移处理

以集群系统配置SUNDB时增加启动如下进程

- cdispatcher： 负责集群数据包的发送/接收及会话管理等
- cserver： 对其所属的成员节点的数据库执行更改和查询操作

SUNDB集群系统需要成员节点之间进行复杂的集群协议通信集群调度器（cdispatcher）能有效地执行网络通信上下文管理及数据包分发机制此外还可通过Heartbeat等持续监视集群会话的有效性

在集群系统中会出现特定节点（驱动程序节点）中执行的SQL需要通过参考目标表的分片策略在远程成员节点上存储数据或查询在相应节点中存储的数据的情况这种情况下每个成员节点都需要一个进程来处理从远程来的该请求并返回结果执行该任务的进程就是cserver

客户端进程

与单机版相同在集群系统中用户也可以使用客户端服务器模型（以下称‘C/S’）和直接访问模型（以下称‘D/A’）但由于集群系统的每个成员节点都有属于自己的监听器因此客户端程序必须预先认知自己要访问的集群成员的监听端口（在C/S模式下）不管访问集群系统的哪个节点用户均可以像单机版数据库一样处理各种类型的事务

此外在单机版和集群系统中以相同的方式处理Signal Handling清理连接和释放各种共享资源等

实例的内存结构

SUNDB集群系统是将多个Shared-Nothing数据库绑定为单个集群系统的管理单元因此各集群成员节点的内存使用方法与单机版系统的类似每个成员节点使用的内存大小由其参数文件中指定的相关参数决定静态区域与单机版相同包含用于数据库管理的实例的基本信息及各会话statement和事务信息重做日志缓冲区字典缓存以及其他各种运行信息另外除了存储基本的管理信息外还存储位置信息和集群会话信息等用于管理集群系统的其他信息

表空间区域由包含各表空间内容的页面框架与用于控制的页面控制标题（Page Control HeaderPCH）组成

应用程序进程内存包含连接时附加的实例内存以进程为单位共享的ODBC环境与各种ODBC句柄以及包含绑定信息等其他信息的堆内存区域

启动与结束集群系统

要启动SUNDB系统应事先在每个成员节点上创建一个实例（gcreatedb）并注册集群组和集群组成员之后可使用sysdba角色通过gsql及gsqlnet启动或结束系统

如果要在C/S模型的专用模式下启动或结束SUNDB集群系统即使用gsqlnet时必须先打开**listener**

在C/S模型的共享模式下无法启动或结束SUNDB 集群系统

```
% gsql --as sysdba
```

```
Enter user-name: sys
Enter password:

Connected to an idle instance.

gSQL>
```

SUNDB集群系统具有以下启动阶段与单机版系统相比OPEN状态细分为LOCAL OPEN与GLOBAL OPEN

- NOMOUNT
  - 打开管理SUNDB实例的守护进程gmaster进程
- MOUNT
  - 使用$ SUNDB_DATA环境变量读取参数和用于恢复的控制文件
- LOCAL OPEN
  - 从数据文件加载表空间的内容后使用重做日志文件进行恢复恢复In-Doubt全局事务新建No-Logging索引并创建字典缓存
- OPEN（GLOBAL OPEN）
  - 连接集群成员之间的集群会话组织分片映射(Shard Map)等选择全局协调器（Global Coordinator）和组协调器（Group Coordinator）最后等待用户的服务接入

启动或结束SUNDB集群系统应通过以下预先操作配置集群系统环境如果使用gcreatedb创建各个成员数据库时给予了唯一的成员名称主机地址端口号则可以省略每个节点的属性设置过程

- 参数设置： 修改要在每个成员节点使用的参数文件

- 创建数据库： 在各个成员节点上通过gcreatedb创建数据库
- 创建集群组及添加成员： 添加构成集群系统的组及成员
- 启动监听器： 为了使用gsqlnet必须先启动每个节点的监听器

使用以下语句添加集群组和成员：

- **ALTER CLUSTER GROUP name ADD MEMBER**
- **CREATE CLUSTER GROUP**
- 创建数据库：在各个节点执行

```
% gcreatedb --cluster --db_name='sundb' --member='g1n1' \
    --host='192.168.0.11' --port 10110
  % gcreatedb --cluster --db_name='sundb' --member='g1n2' \
    --host='192.168.0.12' --port 10120
```

- 创建集群组及成员：在一个节点执行创建以及添加的集群成员的startup阶段应为GLOBAL OPEN

```
gSQL> create cluster group g1 cluster member g1n1
        host '192.168.0.11' port 10110;
  gSQL> alter cluster group g1 add cluster member g1n2
        host '192.168.0.12' port 10120;
```

按照上述完成配置SUNDB集群系统后可以使用以下两种方法启动或结束整个集群系统

- 连接到各个成员节点并逐个启动或关闭
  - 使用\startup及\shutdown命令

- 在一个成员节点同时启动或关闭所有成员节点
  - 通过gsqlnet连接后使用\cstartup及\cshutdown命令

以下介绍按照上述第一个方法连接到各个成员节点后启动集群系统的示例\startup命令用于在没有中间阶段的情况下直接进入LOCAL OPEN阶段它可以分以下三个步骤执行：\startup nomountalter system mount database和alter system open local database

当每个节点通过\startup启动到LOCAL OPEN阶段后最终连接到一个节点并启动到GLOBAL OPEN阶段即可

- 启动到LOCAL OPEN阶段：在各个成员节点执行

```
% gsql sys gliese --as sysdba
    gSQL> \startup
    Startup success.
```

- 启动到OPEN：在一个节点执行

```
% gsql sys gliese --as sysdba
  gSQL> alter system open global database;
  System altered.
```

如果集群系统中包含大量节点通过上述第一种方法启动可能会给操作人员造成压力因为集群系统结束后再次启动时每次启动时都要在每个成员节点上执行从集群系统结束到LOCAL OPEN的整个过程为了方便操作可以通过下面的简单方法一次启动所有成员

- 启动到GLOBAL OPEN阶段：在一个节点执行

```
% gsqlnet sys gliese --as sysdba

  gSQL> \cstartup

  Startup success.
```

> Note:
>
> * \cstartup和\cshutdown命令仅在gsqlnet中可用在gsql中不支持另外启动成员节点时必须提前激活所有成员节点的监听器
>
> * 在单个物理节点上构建包含多个成员的SUNDB集群系统时需要注意创建各个数据库时主目录及成员名称集群端口信息不得出现重复

SUNDB系统结束后管理守护程序进程gmaster将在所有成员节点上终止从而无法再执行连接及其他数据库操作

> Note:
>
> 在SUNDB cluster system中ABORT是唯一可以用于只终止单一节点的终止模式 在ABORT模式下与连接中的session状态无关立即关闭gmaster并删除实例

如下如果使用gsql执行\shutdown命令则只终止连接的成员节点

```
% gsql sys gliese --as sysdba


Connected to SUNDB Database.


gSQL> \shutdown abort
```

```
Shutdown success

gSQL>
```

要一次终止集群系统中的所有成员时可如下使用gsqlnet的\cshutdown命令普通选项可省略

```
% gsqlnet sys gliese --as sysdba

Connected to SUNDB Database.

gSQL> \cshutdown normal

Shutdown success

gSQL>
```

> Caution:
>
> 在执行\cshutdown时使用abort选项的情况下将强制结束所有成员节点的gmaster因此使用\cstartup命令重新启动时根据结束时的节点状态部分成员节点可能无法加入集群系统虽然失败的节点可以通过以下Join命令及重新平衡（Rebalance）过程再次加入但是除特殊情况外建议尽量使用\ cshutdown normal命令结束会更安全

要结束或启动集群系统中的一个成员节点或部分成员节点时按照如下过程进行操作以下为在集群系统中仅重启G1N2成员节点并使其再次加入集群系统的过程

```
% gsql sys gliese --as sysdba --dsn=g1n2


Connected to SUNDB Database.


gSQL> \shutdown abort


Shutdown success


gSQL> \startup


Startup success


gSQL> alter system join database;


System altered.
```

在产生事务的情况下结束成员节点后重启该节点并要再次加入集群系统时需要对已修改的表进行重新平衡（rebalancing）操作如果不执行重新平衡操作则在驱动程序节点上执行的事务可能无法修改未执行重新平衡的表

重新平衡操作是指在成员节点之间同步表的数据分发策略及属性信息并根据数据分发策略将表数据再次划分存储在各个成员节点的整个过程

```
% gsql sys gliese --as sysdba


Connected to SUNDB Database.
```

```
gSQL> \shutdown abort

Shutdown success

gSQL> \startup

Startup success

gSQL> alter system join database;

ERR-42000(16405): some tables in the database need to be rebalanced

System altered.

gSQL> alter database rebalance;

Database altered.
```

3.2 安装SUNDB与创建数据库

安装及创建要包含在SUNDB集群系统中的成员数据库的方法与单机版大致相同本章仅介绍与单机版系统相比集群系统独有的安装及创建数据库的特征与方法

配置SUNDB安装包

用于配置SUNDB集群系统的安装包与配置单机版数据库的安装包相同但是创建字典及性能视图等的脚本分为单机版系统脚本和集群系统脚本因此用户应在创建数据库后使用对应的脚本进行创建

单机版数据库的脚本位于$SUNDB_HOME/admin/standalone目录下集群系统的脚本位于$SUNDB_HOME/admin/cluster目录下

文件名	说明
README	read me
DictionarySchema.sql	字典模式创建脚本
InformationSchema.sql	信息模式创建脚本
PerformanceViewSchema.sql	性能视图(Performance View)模式创建脚本

Table 3-2 admin/standalone目录

文件名	说明
README	read me
DictionarySchema.sql	字典模式创建脚本
InformationSchema.sql	信息模式创建脚本
PerformanceViewSchema.sql	性能视图(Performance View)模式创建脚本

Table 3-3 admin/cluster目录

安装SUNDB软件

应在SUNDB系统包含的所有成员节点上安装SUNDB软件安装方法与单机版系统相同设置及查看内核参数的方法设置环境变量的方法与单机版系统相同可参考相应的教程

在这种情况下注意各个集群成员节点要设置相同的参数设置或数据库名称数据库版本字符集时区等才可以配置正常的集群系统

创建数据库

与单机版系统相同使用gcreatedb工具在配置集群系统的每个成员节点上创建数据库

以下gcreatedb选项仅在创建集群数据库时使用其他选项与单机版系统相同

执行参数	说明
--cluster	表示集群数据库 如果被省略则将创建单机版数据库
--member	在集群系统中使用的本地数据库的成员名称 如果被省略则将使用LOCAL_CLUSTER_MEMBER参数中设置的值
--host	用于集群系统成员之间进行通信的本地成员的IP地址 如果被省略则将使用LOCAL_CLUSTER_MEMBER_HOST参数中设置的值
--port	用于集群系统成员之间进行通信的本地成员的TCP监听端口 如果被省略则使用LOCAL_CLUSTER_MEMBER_PORT参数中设置的值

Table 3-4 gcreatedb参数

以下为创建在集群系统中使用的数据库的示例

```
[SHELL]> gcreatedb --cluster

Database created


[SHELL]> gcreatedb --cluster --member=G1N1

Database created


[SHELL]> gcreatedb --cluster --member=G1N1 --host=127.0.0.1 --port 10101

Database created


[SHELL]> gcreatedb --cluster                          \
```

```
                    --db_name="TEST_DB"                \
                    --home=$SUNDB_DATA            \
                    --host=127.0.0.1                   \
                    --port=10101                       \
                    --db_comment="g1n1 db comment"     \
                    --timezone="+09:00"                \
                    --character_set="UHC"              \
                    --char_length_units="OCTETS"
Database created


[SHELL]> ls $SUNDB_DATA/db
system_data.dbf  system_dict.dbf  system_trans.dbf system_undo.dbf
```

构建字典模式(Dictionary Schema)信息

与单机版系统相同为了正常使用集群系统需要构建字典模式信息如果不构建以下模式则获取对象结构信息的ODBCJDBC的目录API（例如：SQLTables（）函数等）会出现误操作而可能无法与第三方工具互通因此创建数据库后必须构建字典模式

建议在集群系统中完成添加集群群组和成员的操作之后构建模式因为完成集群系统配置后连接到一个成员节点执行创建脚本时SUNDB会自动在所有成员节点上创建模式

- DICTIONARY_SCHEMA：由查询DBA_ *ALL_ *USER_ *等对象信息的视图及表组成
- INFORMATION_SCHEMA：由属于SQL标准的INFORMATION_SCHEMA中包含的视图及表组成

- PERFORMANCE_VIEW_SCHEMA：由通过组合固定表（fixed table）的信息来查询系统信息的视图组成

以下介绍了使用集群系统的脚本构建模式的方法如上所述通过接入GLOBAL OPEN阶段中的一个成员节点执行即可

```
% gsql sys gliese --as sysdba --import
$SUNDB_HOME/admin/cluster/DictionarySchema.sql
% gsql sys gliese --as sysdba --import
$SUNDB_HOME/admin/cluster/InformationSchema.sql
% gsql sys gliese --as sysdba --import
$SUNDB_HOME/admin/cluster/PerformanceViewSchema.sql
```

可使用上述构建的模式信息查询集群系统的各种结构信息与单机版系统不同的是查询时可通过在对象之后给予群组和成员名称来提取用户想要查询的节点信息

```
% gsql test test

Connected to SUNDB Database.

gSQL> select origin_member_name, stat_name, stat_value
    2   from gv$system_mem_stat
    3  where stat_name = 'PLAN_CACHE_TOTAL_SIZE';

ORIGIN_MEMBER_NAME STAT_NAME             STAT_VALUE
------------------ --------------------- ----------
```

```
G1N1                PLAN_CACHE_TOTAL_SIZE    17844952

G2N2                PLAN_CACHE_TOTAL_SIZE    16796288

G2N1                PLAN_CACHE_TOTAL_SIZE    16796288

G1N2                PLAN_CACHE_TOTAL_SIZE    16796288

G3N1                PLAN_CACHE_TOTAL_SIZE    16796288

G3N2                PLAN_CACHE_TOTAL_SIZE    16796288


6 rows selected.


gSQL> select origin_member_name, stat_name, stat_value
    2   from gv$system_mem_stat@g1n2
    3  where stat_name = 'PLAN_CACHE_TOTAL_SIZE';


ORIGIN_MEMBER_NAME STAT_NAME             STAT_VALUE

------------------ --------------------- ----------

G1N2                PLAN_CACHE_TOTAL_SIZE    16796288


1 row selected.


gSQL> select origin_member_name, stat_name, stat_value
    2   from gv$system_mem_stat@g1
    3  where stat_name = 'PLAN_CACHE_TOTAL_SIZE';


ORIGIN_MEMBER_NAME STAT_NAME             STAT_VALUE

------------------ --------------------- ----------
```

```
G1N1                PLAN_CACHE_TOTAL_SIZE    17844952

G1N2                PLAN_CACHE_TOTAL_SIZE    16796288


2 row selected.
```

3.3 管理模式对象

模式对象是用户创建的逻辑结构SUNDB集群系统支持表索引全局序列（Global Sequence）等模式对象本章介绍与单机版系统相比的集群系统的各个模式对象的特征以及有效管理其模式对象的方法

表管理

在SUNDB集群系统中创建表时用户可以选择指定以下四种分片策略中的一种分片策略是指将表数据分配并存储到集群系统中的各个集群群组的方式此选项只能在集群系统中指定无法在单机版数据库中使用

- Cloned strategy
  - 同等地复制表中的所有数据
- Hash sharding strategy
  - 以分片键的Hash值为准分配表数据
- Range sharding strategy
  - 以分片键的范围（Range）值为准分配表数据
- List sharding strategy
  - 以分片键的列表（List）值为准分配表数据

创建表的详细内容请参考**CREATE TABLE** 命令和 **集群表与分片**

如果省略“分片策略”选项则由**DEFAULT_SHARDING**参数决定DEFAULT_SHARDING的默认值为0且创建克隆表

```
CREATE TABLE t1
(
    id   INTEGER,
    name VARCHAR(128)
);
```

如上如果用户创建表时未给予分片策略则SUNDB集群系统将使用以下语句在内部创建表

```
CREATE TABLE t1
(
    id   INTEGER,
    name VARCHAR(128)
)
    CLONED
    AT CLUSTER WIDE
;
```

> Note:
>
> Hash/Range/List分片表在创建约束条件时必须符合以下条件：
>
> - 主键唯一约束必须包含分片键

Cloned Strategy

并非根据特定条件划分表数据而是一种复制所有数据的方式复制数据的对象可以是集群系统中的所有节点也可以是特定的集群群组 即在指定的集群群组的集群成员中放置副本（clone）的策略

- AT CLUSTER WIDE
  - 在集群系统中的所有集群群组的所有集群成员中放置克隆
  - 添加集群群组集群成员时可以使用**ALTER TABLE name REBALANCE** 语句重新放置克隆
- AT CLUSTER GROUP group_list
  - 在指定集群群组的所有集群成员中放置克隆
  - 在指定的集群群组中添加集群成员时可以通过 **ALTER TABLE name REBALANCE**语句来重新放置克隆
  - 添加集群群组不会影响克隆的重新放置

如果省略该选项则默认值将自动设置为AT CLUSTER WIDE

```
CREATE TABLE t1
(
    id   INTEGER,
    name VARCHAR(128)
)
CLONED
AT CLUSTER WIDE;
```

```
CREATE TABLE t1
(
    id   INTEGER,
    name VARCHAR(128)
```

```
)
CLONED
AT CLUSTER GROUP G1, G2;
```

Hash Sharding Strategy

是一种指根据定义为分片键的列的Hash值来分配表数据的策略

使用hash分片策略需要按照以下条件指定分片键

- 最多可列出32列
- 不能使用重复的列
- 不能使用 LONG VARCHARLONG VARBINARY类型的列

```
CREATE TABLE t1 ( id INTEGER, name VARCHAR(128) )
    SHARDING BY HASH(id);
```

如果省略与上述Hash分片相关的选项SUNDB系统将其解释为如下语句

```
CREATE TABLE t1 ( id INTEGER, name VARCHAR(128) )
    SHARDING BY HASH(id)
    SHARD COUNT 24
    AT CLUSTER WIDE;
```

根据id列的hash值将表的行分配到24个分片中的一个并且在整个集群系统中均匀分配24个分片

更多详细信息请参考**集群表与分片**

Range Sharding Strategy

是一种根据指定为分片键的列的范围值分配表数据的策略根据各范围值来区分的分片可以指定特定群组后分配也可以按CLUSTER WIDE进行分配

以下为基于分片键列值的范围定义六个范围分片后为了按CLUSTER WIDE进行分配而使用的表创建语句 如果在创建表后添加了集群群组则可以使用REBALANCE功能来执行包含添加群组在内的分片的重新分配操作

- 创建范围分片表
- 在现有的集群群组（g1g2g3）中分配6个分片

```
CREATE TABLE t1
(
   id   INTEGER,
   name VARCHAR(32)
)
SHARDING BY RANGE (id)
    AT CLUSTER WIDE
    SHARD s1 VALUES LESS THAN ( 200000 ),
    SHARD s2 VALUES LESS THAN ( 400000 ),
    SHARD s3 VALUES LESS THAN ( 500000 ),
    SHARD s4 VALUES LESS THAN ( 600000 ),
    SHARD s5 VALUES LESS THAN ( 800000 ),
    SHARD s6 VALUES LESS THAN ( MAXVALUE )
;
```

- 添加集群群组

```
CREATE CLUSTER GROUP g4
       CLUSTER MEMBER g4n1 HOST '192.168.0.41' PORT 10401
;
```

- 重新分配range shard
- 将6个分片重新分配到包含已创建的g4在内的（g1g2g3g4）集群群组中

```
ALTER TABLE t1 REBALANCE;
```

以下是将根据分片键列值的范围定义的范围分片仅分配于特定集群群组的语句示例即范围值小于200000的shard s1分配到集群群组g1s2分配到集群群组g2s3分配到集群群组g3如此根据各个范围分片指定集群群组后生成的表即使后续添加集群群组也可以通过REBALANCE功能重新分配分片

- 创建范围分片表
- 将各个范围分片分配到指定集群群组

```
CREATE TABLE t1
(
   id   INTEGER,
   name VARCHAR(32)
)
SHARDING BY RANGE (id)
    SHARD s1 VALUES LESS THAN ( 200000 )   AT CLUSTER GROUP g1,
    SHARD s2 VALUES LESS THAN ( 400000 )   AT CLUSTER GROUP g2,
```

```
    SHARD s3 VALUES LESS THAN ( 500000 )   AT CLUSTER GROUP g3,
    SHARD s4 VALUES LESS THAN ( 600000 )   AT CLUSTER GROUP g2,
    SHARD s5 VALUES LESS THAN ( 800000 )   AT CLUSTER GROUP g3,
    SHARD s6 VALUES LESS THAN ( MAXVALUE ) AT CLUSTER GROUP g1
;
```

- 添加集群群组

```
CREATE CLUSTER GROUP g4
       CLUSTER MEMBER g4n1 HOST '192.168.0.41' PORT 10401
;
```

- 重新分配范围分片
- 新创建的集群群组g4中未分配分片

```
ALTER TABLE t1 REBALANCE;
```

指定范围分片键时应考虑以下条件：

- 最多可列出32列
- 不能使用相同的列
- 不能使用LONG VARCHARLONG VARBINARY类型的列

List Sharding Strategy

是一种根据定义为分片键的列的列表（List）值分配表数据的策略与范围分片策略相同每个分片可以按CLUSTER WIDE进行分配也可以通过将各分片指定到特定群组进行分配

在列表分片策略中指定分片键的条件如下：

- 只能使用一列
- 不能使用LONG VARCHARLONG VARBINARY类型的列

以下是创建表的语句示例其中创建的列表分片可按CLUSTER WIDE进行分配可以使用REBALANCE功能重新分配包含创建表后添加的集群群组在内的分片

- 创建list sharded表
- 在现有的集群群组（g1g2g3）中分配5个分片

```
CREATE TABLE city
(
   id   INTEGER,
   name VARCHAR(32)
)
SHARDING BY LIST (name)
    AT CLUSTER WIDE
    SHARD s1 VALUES IN ( 'SEOUL' ),
    SHARD s2 VALUES IN ( 'PUSAN', 'ULSAN', 'DAEGU' ),
    SHARD s3 VALUES IN ( 'DAEJEON', 'GWANGJU' ),
    SHARD s4 VALUES IN ( 'ANSAN', 'GOYANG' ),
    SHARD s5 VALUES IN ( DEFAULT )
;
```

- 添加集群群组

```
CREATE CLUSTER GROUP g4
       CLUSTER MEMBER g4n1 HOST '192.168.0.41' PORT 10401
;
```

- 重新分配list shard
- 将5个分片重新分配到包含添加的g4在内的(g1, g2, g3, g4) 集群群组中

```
ALTER TABLE city REBALANCE;
```

以下是将根据分片键列值的排列值定义的列表分片分配到特定集群群组的语句示例即使后续添加集群群组也无法使用REBALANCE功能重新分配分片

- 创建list sharded表
- 将各个列表分片分配至指定集群群组

```
CREATE TABLE city
(
   id   INTEGER,
   name VARCHAR(32)
)
SHARDING BY LIST (name)
    SHARD s1 VALUES IN ( 'SEOUL' )                   AT CLUSTER GROUP g1,
    SHARD s2 VALUES IN ( 'PUSAN', 'ULSAN', 'DAEGU' ) AT CLUSTER GROUP g2,
    SHARD s3 VALUES IN ( 'DAEJEON', 'GWANGJU' )      AT CLUSTER GROUP g3,
    SHARD s4 VALUES IN ( 'ANSAN', 'GOYANG' )         AT CLUSTER GROUP g2,
    SHARD s5 VALUES IN ( DEFAULT )                   AT CLUSTER GROUP g1
```

```
;
```

- 添加集群群组

```
CREATE CLUSTER GROUP g4
        CLUSTER MEMBER g4n1 HOST '192.168.0.41' PORT 10401
;
```

- 重新分配list shard
- 创建的集群群组g4中不分配分片

```
ALTER TABLE city REBALANCE;
```

索引管理

全局二级索引

集群系统中存在多个成员节点并且根据分片策略将表记录划分或复制存储到各个成员节点单机版系统在存储记录时在其数据库中同时存储唯一值（行标识符：RID）从而保证记录的唯一性但是在集群系统中每个节点都可以具有重复值因此该值无法在集群系统中保证唯一性

因此需要在整个集群系统中保障记录的唯一性为此添加了全局RID（GRID）即使更新记录也不会变更记录的GRID值分片键变更并移动到其他分片也不会变更因此可在集群系统环境下保证特定记录的唯一性

全局二级索引是为了在集群系统环境下可快速检索记录的GRID值而通过密钥组成的B-Tree索引

用户创建表时可通过**DEFAULT_GLOBAL_SECONDARY_INDEX_CREATION**选择是否创建全局二级索引完成表创建后可删除或重新创建全局二级索引 每个表只能创建一个全局二级索引

为了执行表的非确定性查询必须要有全局二级索引如果表中不存在全局二级索引则非确定性查询将失败如下所示：

```
gSQL> DELETE FROM T1 LIMIT 1;


ERR-42000(16423): does not support non-deterministic DML in the cluster
system : global secondary index expected
```

确认表是否已创建全局二级索引可以如下查询USER_GSI_PLACE字典或查询ALL_GSI_PLACE及DBA_GSI_PLACE字典

```
gSQL> CREATE TABLE T1( I1 INTEGER );


Table created.


gSQL> COMMIT;


Commit complete.


gSQL> SELECT *
    2   FROM USER_GSI_PLACE@LOCAL
    3  WHERE TABLE_NAME = 'T1';
```

```
TABLE_SCHEMA TABLE_NAME GROUP_ID GROUP_NAME MEMBER_ID MEMBER_NAME

MEMBER_OFFLINE

------------ ---------- -------- ---------- --------- ----------- --------

------

BLOCKS

------

PUBLIC       T1                1 G1                 1 G1N1        FALSE

    64

PUBLIC       T1                1 G1                 2 G1N2        FALSE

  null


2 rows selected.


gSQL> DROP TABLE T1;


Table dropped.


gSQL> COMMIT;


Commit complete.


gSQL> SELECT *

    2   FROM USER_GSI_PLACE@LOCAL

    3  WHERE TABLE_NAME = 'T1';
```

```
no rows selected.
```

全局序列

SUNDB集群系统为了使多个成员节点共享使用符合用户指定条件的序列值的集合为用户提供了扩展现有序列对象的全局序列对象即当用户在集群系统中创建序列时内部会自动创建全局序列对象当在每个成员节点上调用NEXTVAL时从该全局对象分配特定范围的序列值并使用如下创建及使用全局序列对象的方法与单机版系统相同

```
gSQL> CREATE SEQUENCE global_user_seq START WITH 1000 INCREMENT BY 1
NOCACHE NOCYCLE;

Sequence created.

gSQL> SELECT global_user_seq.NEXTVAL FROM dual;

NEXTVAL
-------
      1

1 row selected.

gSQL> DROP SEQUENCE global_user_seq;

Sequence dropped.
```

与单机版使用的序列对象相同全局序列对象可以在创建选项中指定缓存大小这意味着能否从全局序列对象中获取多个序列值并将其加载到本地高速缓存中下面是将缓存大小设置为5时创建全局序列对象后的每个成员节点的本地缓存状态及调用NEXTVAL时的返回值

```
gSQL> CREATE SEQUENCE seq START WITH 1 CACHE 5;


Sequence created.
```

成员名称	NEXTVAL结果	本地缓存序列剩余数量		说明
		G1N1	G1N2	
G1N1	1	4	0	From Global Object (Alloc 1 ~ 5)
G1N1	2	3	0	From Local Cache
G1N1	3	2	0	From Local Cache
G1N2	6	3	4	From Global Object (Alloc 6 ~ 10)
G1N2	7	3	3	From Local Cache

成员名称	NEXTVAL结果	本地缓存序列剩余数量		说明
		G1N1	G1N2	
G1N2	8	3	2	From Local Cache
G1N2	9	3	1	From Local Cache
G1N2	10	3	0	From Local Cache
G1N2	11	3	4	From Global Object (Alloc 11 ~ 15)
G1N2	12	3	3	From Local Cache
G1N1	4	1	3	From Local Cache
G1N1	5	0	3	From Local Cache
G1N1	16	4	3	From Global Object (Alloc 16 ~ 20)

从上表中可以看出G1N1和G1N2分别从全局序列对象各分配到了两次共4次的序列值由于在创建

序列时将缓存选项的值设置为5因此当全局序列对象分配时G1N1和G1N2各自分配到五个序列值此时成员节点将分配到的五个序列值存储在自己的本地缓存中之后在每次调用NEXTVAL时逐个进行返回

- G1N1
  - 当第一次调用NEXTVAL时从全局序列对象分配到五个（1~5）序列值
  - 作为NEXTVAL的结果返回1个序列且其余四个序列值被加载到自己的本地缓存中
  - 之后调用NEXTVAL时按顺序依次返回加载在本地缓存中的序列值
- G1N2
  - 当第一次调用NEXTVAL时分配到G1N1已分配到的范围之后的序列值（6~10）
  - 作为NEXTVAL的结果返回6并且剩余的四个序列值（7~10）加载到自己的本地缓存中
  - 之后调用NEXTVAL时返回加载在本地缓存中的序列值
  - 本地缓存耗尽后再次从全局序列对象分配到五个（11~15）序列值
- G1N1
  - 本地缓存耗尽后分配到G1N2最后分配到的序列值之后的五个（16~20）序列值

如果从未在成员节点上调用过NEXTVAL或已耗尽分配到的序列值则成员节点必须从全局序列对象中分配到与缓存大小相同的序列因此当系统需快速获取序列值时要在创建序列时设置适当大小的缓存以防止产生过多的频繁分配因为与单机版数据库不同从全局序列对象分配序列值时会产生网络通信成本

由于系统故障或管理员操作等原因重新启动数据库此时无法重新使用本地缓存中加载的序列值重新启动后 第一次调用NEXTVAL时会从全局序列对象重新分配序列值 因此在创建序列时设置的缓存大小也意味着当故障发生时可能丢失的序列的范围所以设置大小时不仅要考虑响应性能还要考虑丢失范围

在数据库集群系统中以特定节点为准调用的NEXTVAL的结果值可能是不连续的在上面的示例中可以看到G1N1节点在调用NEXTVAL时返回的序列值是不连续的在耗尽第一次分配到的序列值（1~5）之后第二次分配到的序列值是16~20因此将16作为5的下一个序列值返回给用户这是由于多个节点竞争性的分配使用一个全局序列池导致的

与现有的单机版数据库的序列对象相比全局序列对象的特征及约束事项总结如下：

- 在ALTER SEQUENCE语句中不能使用INCREMENT BY选项更改符号（可以修改大小）
- 使用CYCLE选项时根据整体序列池的大小成员节点之间会返回重复值因此如果需要CYCLE选项则应考虑到INCREMENT BYCACHE SIZE及集群成员节点的数量创建足够大的序列池
- 即使非故障情况基于特定成员节点返回的序列值也可能是不连续的当然如果只有一个成员节点调用NEXTVAL则可以获得连续的序列值
- 在NOCACHE的情况下由于CACHE SIZE为1因此本地缓存中不存储多余的序列值这意味着每次调用NEXTVAL时都会从全局序列对象中逐个分配序列值而增加网络成本性能可能会降低
- 一般使用AUTO COMMIT进行创建更新和删除序列对象
- 通过ALTER语句更改CACHE及INCREMENT大小时将重置所有节点本地高速缓存中加载的序列值即后续调用NEXTVAL时要重新从全局序列对象分配新的序列值

3.4 SUNDB参数

以下为SUNDB集群系统中可使用的主要参数

名称	说明
LOCAL_CLUSTER_MEMBER	成员名称
LOCAL_CLUSTER_MEMBER_HOST	用于连接集群会话的主机地址
LOCAL_CLUSTER_MEMBER_PORT	用于连接集群会话的端口
CDISPATCHER_THREADS	集群调度程序线程数量
CSERVERS	集群服务器进程数量
CLUSTER_DATA_SYNC_SERVERS	用于同步副本数据的集群服务器进程数量

在SUNDB集群系统中每个参数都具备以下两种可更改的范围（Scope）

- LOCAL： 可以更改全体成员的参数值或更改特定成员的参数值
- GLOBAL： 不能更改特定成员的参数值并且更改必须应用于集群系统中的所有成员

例如PRIVATE_STATIC_AREA_SIZE参数如下所示可更改至LOCAL范围（IS_GLOBAL列= FALSE）

因此可通过alter system set语句更改全体成员节点或指定特定成员进行更改如果在alter system set语句后未添加AT子句则更改的参数将应用于集群系统的所有成员

```
% gsql test test


Connected to SUNDB Database.
```

```
gSQL> select origin_member_name, property_name, property_value, is_global
    2   from gv$property
    3  where property_name = 'PRIVATE_STATIC_AREA_SIZE';

ORIGIN_MEMBER_NAME PROPERTY_NAME            PROPERTY_VALUE IS_GLOBAL
------------------ ------------------------ -------------- ---------
G1N1               PRIVATE_STATIC_AREA_SIZE 104857600      FALSE
G1N2               PRIVATE_STATIC_AREA_SIZE 104857600      FALSE

2 rows selected.

gSQL> alter system set private_static_area_size = 200000000 at g1n2;

System altered.

gSQL> select origin_member_name, property_name, property_value, is_global
    2   from gv$property
    3  where property_name = 'PRIVATE_STATIC_AREA_SIZE';

ORIGIN_MEMBER_NAME PROPERTY_NAME            PROPERTY_VALUE IS_GLOBAL
------------------ ------------------------ -------------- ---------
G1N1               PRIVATE_STATIC_AREA_SIZE 104857600      FALSE
G1N2               PRIVATE_STATIC_AREA_SIZE 200000000      FALSE
```

```
2 rows selected.


gSQL> alter system set private_static_area_size = 300000000;


System altered.


gSQL> select origin_member_name, property_name, property_value, is_global
    2   from gv$property
    3  where property_name = 'PRIVATE_STATIC_AREA_SIZE';


ORIGIN_MEMBER_NAME PROPERTY_NAME            PROPERTY_VALUE IS_GLOBAL
------------------ ------------------------ -------------- ---------
G1N1               PRIVATE_STATIC_AREA_SIZE 300000000      FALSE
G1N2               PRIVATE_STATIC_AREA_SIZE 300000000      FALSE


2 rows selected.
```

相反如下所示DDL_AUTOCOMMIT可变更的范围为GLOBAL因此在alter system set语句中使用AT子句指定特定成员时会发生错误

```
% gsql test test


Connected to SUNDB Database.


gSQL> select origin_member_name, property_name, property_value, is_global
    2   from gv$property
```

```
    3  where property_name = 'DDL_AUTOCOMMIT';


ORIGIN_MEMBER_NAME PROPERTY_NAME  PROPERTY_VALUE IS_GLOBAL

------------------ -------------- -------------- ---------

G1N1               DDL_AUTOCOMMIT NO             TRUE

G1N2               DDL_AUTOCOMMIT NO             TRUE


2 rows selected.


gSQL> alter system set ddl_autocommit = false at g1n2;


ERR-42000(16398): the domain of property does not match with domain

'G1N2' :

alter system set ddl_autocommit = false at g1n2

                                           *

ERROR at line 1:


gSQL> alter system set ddl_autocommit = false;


System altered.


gSQL> select origin_member_name, property_name, property_value, is_global

    2   from gv$property

    3  where property_name = 'DDL_AUTOCOMMIT';

ORIGIN_MEMBER_NAME PROPERTY_NAME  PROPERTY_VALUE IS_GLOBAL
```

```
------------------ -------------- -------------- ---------

G1N1               DDL_AUTOCOMMIT NO             TRUE

G1N2               DDL_AUTOCOMMIT NO             TRUE


2 rows selected.
```

4.What's New

4.1 特征矩阵

本章简要介绍每个主要版本添加的主要功能

架构

系统架构

以下为系统架构的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
Shared Nothing Cluster	X	O	O	O	O
D/A (Direct Attach)	O	O	O	O	O
JDBC D/A (Direct Attach)	X	O	O	O	O
C/S (Client/Server) Dedicated	O	O	O	O	O
C/S (Client/Server) Shared	O	O	O	O	O
multi-process applications	O	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
multi-threaded applications	O	O	O	O	O
Linux platform	O	O	O	O	O
HP platform	O	O	O	O	O
AIX platform	O	O	O	O	O
Windows Client Platform	O	O	O	O	O
CDC(Change Data Capture) replication	O	O	O	O	O
CDC replication with log mirror	O	O	O	O	O
multi-level start up	O	O	O	O	O
parallel database loading	O	O	O	O	O
parallel index build	O	O	O	O	O
SQL plan cache	O	O	O	O	O
IPC	X	X	X	O	O

Table 4-1 系统架构的特征矩阵

内部存储(Storage Internal)

以下为内部存储的特征矩阵

特征	2.x	3.x	V5.020	V5.021	V5.022
memory dictionary tablespace	O	O	O	O	O
memory data tablespace	O	O	O	O	O
memory undo tablespace	O	O	O	O	O
memory temporary tablespace	O	O	O	O	O
memory bitmap data segment	O	O	O	O	O
memory bitmap undo segment	O	O	O	O	O
memory bitmap instant segment	O	O	O	O	O
memory heap table	O	O	O	O	O
memory instant table	O	O	O	O	O
memory B-tree index	O	O	O	O	O
memory instant B-tree	O	O	O	O	O
memory instant hash	O	O	O	O	O
global secondary index	X	O	O	O	O
disk data tablespace	X	X	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
disk bitmap data segment	X	X	O	O	O
disk B-tree index	X	X	O	O	O
disk global secondary index	X	X	O	O	O

Table 4-2 内部存储的特征矩阵

事务控制(Transaction Control)

以下为事务控制的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
CDS(Concurrency Data Store) database mode	O	O	O	O	O
TDS(Transactional Data Store) database mode	O	O	O	O	O
read-only database	O	O	O	O	O
read/write database	O	O	O	O	O
flat transaction	O	O	O	O	O
distributed transaction	O	O	O	O	O
read-only transaction	O	O	O	O	O
read/write transaction	O	O	O	O	O
READ COMMITTED isolation level	O	O	O	O	O
SERIALIZABLE isolation level with SELECT FOR UPDATE	O	O	O	O	O
MVCC(Multi Version Concurrency Control)	O	O	O	O	O
multi-version read consistency	O	O	O	O	O
multi-statement consistent read	O	O	O	O	O
implicit lock for DML	O	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
writer don't blocks readers	0	0	0	0	0
row-level locking	0	0	0	0	0
deadlock detection	0	0	0	0	0
deadlock resolution	0	0	0	0	0
lock granularity	0	0	0	0	0
read lock	0	0	0	0	0
write lock	0	0	0	0	0
intention lock	0	0	0	0	0
WAL(Write Ahead Logging)	0	0	0	0	0
repeat history	0	0	0	0	0
restart recovery	0	0	0	0	0
circular logging	0	0	0	0	0
buffered logging	0	0	0	0	0
logging group	0	0	0	0	0
supplemental logging	0	0	0	0	0
mirrored logging	0	0	0	0	0
synchronous commit	0	0	0	0	0

特征	2.x	3.x	V5.020	V5.021	V5.022
asynchronous commit	0	0	0	0	0
grouped commit	0	0	0	0	0
total rollback	0	0	0	0	0
implicit statement rollback	0	0	0	0	0
savepoint management	0	0	0	0	0

Table 4-3 事务控制的特征矩阵

备份和恢复(Backup & Recovery)

以下为备份与恢复的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
off-line backup	O	O	O	O	O
on-line backup	O	O	O	O	O
full backup	O	O	O	O	O
incremental backup	O	O	O	O	O
complete recovery	O	O	O	O	O
incomplete recovery	O	O	O	O	O
auto instance recovery	O	O	O	O	O
tablespace recovery	O	O	O	O	O
file recovery	O	O	O	O	O
change tracking	X	X	O	O	O

Table 4-4 备份与恢复的特征矩阵

数据库信息

DICTIONARY_SCHEMA模式

以下为DICTIONARY_SCHEMA模式的特征矩阵

所属	特征	2.x	3.x	V5.020	V5.021	V5.022
Views of All_family	ALL_ALL_TABLES	O	O	O	O	O
	ALL_ARGUMENTS	X	O	O	O	O
	ALL_CATALOG	O	O	O	O	O
	ALL_CLUSTER_TABLES	X	O	O	O	O
	ALL_COL_COMMENTS	O	O	O	O	O
	ALL_COL_PLACE	X	X	X	X	X
	ALL_COL_PRIVS	O	O	O	O	O
	ALL_COL_PRIVS_MADE	O	O	O	O	O
	ALL_COL_PRIVS_RECD	O	O	O	O	O
	ALL_CONSTRAINTS	O	O	O	O	O
	ALL_CONS_COLUMNS	O	O	O	O	O
	ALL_DB_PRIVS	O	O	O	O	O
	ALL_DB_PRIVS_MADE	O	O	O	O	O

所属	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	ALL_DB_PRIVS_RECD	O	O	O	O	O
	ALL_DEPENDENCIES	X	O	O	O	O
	ALL_GLOBAL_SECONDARY_INDEXES	X	O	O	O	O
	ALL_GSI_PLACE	X	O	O	O	O
	ALL_INDEXES	O	O	O	O	O
	ALL_IND_COLUMNS	O	O	O	O	O
	ALL_IND_PLACE	X	O	O	O	O
	ALL_NONSCHEMA_COMMENTS	O	O	O	O	O
	ALL_OBJECTS	O	O	O	O	O
	ALL_PACKAGE_PRIVS	X	X	O	O	O
	ALL_PACKAGE_PRIV_MADE	X	X	O	O	O
	ALL_PACKAGE_PRIV_RECD	X	X	O	O	O
	ALL_PROCEDURES	X	O	O	O	O
	ALL_PROC_PRIVS	X	O	O	O	O
	ALL_PROC_PRIV_MADE	X	O	O	O	O
	ALL_PROC_PRIV_RECD	X	O	O	O	O
	ALL_SCHEMAS	O	O	O	O	O

所属	特征	2.x	3.x	V5.020	V5.021	V5.022
	ALL_SCHEMA_PATH	O	O	O	O	O
	ALL_SCHEMA_PRIVS	O	O	O	O	O
	ALL_SCHEMA_PRIVS_MADE	O	O	O	O	O
	ALL_SCHEMA_PRIVS_RECD	O	O	O	O	O
	ALL_SEQUENCES	O	O	O	O	O
	ALL_SEQ_PRIVS	O	O	O	O	O
	ALL_SEQ_PRIVS_MADE	O	O	O	O	O
	ALL_SEQ_PRIVS_RECD	O	O	O	O	O
	ALL_SHARD_KEY_COLUMNS	X	O	O	O	O
	ALL_SOURCE	X	O	O	O	O
	ALL_SYNONYMS	O	O	O	O	O
	ALL_TABLES	O	O	O	O	O
	ALL_TAB_COLS	O	O	O	O	O
	ALL_TAB_COLUMNS	O	O	O	O	O
	ALL_TAB_COMMENTS	O	O	O	O	O
	ALL_TAB_IDENTITY_COLS	O	O	O	O	O
	ALL_TAB_PLACE	X	O	O	O	O

所属	特征	2.x	3.x	V5.020	V5.021	V5.022
	ALL_TAB_SHARDS	X	O	O	O	O
	ALL_TAB_PRIVS	O	O	O	O	O
	ALL_TAB_PRIVS_MADE	O	O	O	O	O
	ALL_TAB_PRIVS_RECD	O	O	O	O	O
	ALL_TBS_PRIVS	O	O	O	O	O
	ALL_TBS_PRIVS_MADE	O	O	O	O	O
	ALL_TBS_PRIVS_RECD	O	O	O	O	O
	ALL_USERS	O	O	O	O	O
	ALL_VIEWS	O	O	O	O	O
Views of DBA_family	DBA_ALL_TABLES	O	O	O	O	O
	DBA_ARGUMENTS	X	O	O	O	O
	DBA_CATALOG	O	O	O	O	O
	DBA_CLUSTER	X	O	O	O	O
	DBA_CLUSTER_COMMENTS	X	O	O	O	O
	DBA_CLUSTER_TABLES	X	O	O	O	O
	DBA_COL_COMMENTS	O	O	O	O	O
	DBA_COL_PLACE	X	X	X	X	X

所属	特征	2.x	3.x	V5.020	V5.021	V5.022
	DBA_COL_PRIVS	O	O	O	O	O
	DBA_CONSTRAINTS	O	O	O	O	O
	DBA_CONS_COLUMNS	O	O	O	O	O
	DBA_DB_PRIVS	O	O	O	O	O
	DBA_DEPENDENCIES	X	O	O	O	O
	DBA_EXTENTS	O	O	O	O	O
	DBA_GLOBAL_SECONDARY_INDEXES	X	O	O	O	O
	DBA_GSI_PLACE	X	O	O	O	O
	DBA_INDEXES	O	O	O	O	O
	DBA_IND_COLUMNS	O	O	O	O	O
	DBA_IND_PLACE	X	O	O	O	O
	DBA_NONSCHEMA_COMMENTS	O	O	O	O	O
	DBA_OBJECTS	O	O	O	O	O
	DBA_PACKAGE_PRIVS	X	X	O	O	O
	DBA_PROCEDURES	X	O	O	O	O
	DBA_PROC_PRIVS	X	O	O	O	O
	DBA_PROFILES	O	O	O	O	O

所属	特征	2.x	3.x	V5.020	V5.021	V5.022
	DBA_RECYCLEBIN	X	X	O	O	O
	DBA_SCHEMAS	O	O	O	O	O
	DBA_SCHEMA_PATH	O	O	O	O	O
	DBA_SCHEMA_PRIVS	O	O	O	O	O
	DBA_SEQUENCES	O	O	O	O	O
	DBA_SEQ_PRIVS	O	O	O	O	O
	DBA_SHARD_KEY_COLUMNS	X	O	O	O	O
	DBA_SOURCE	X	O	O	O	O
	DBA_STAT_SYSTEM	X	O	O	O	O
	DBA_SYNONYMS	O	O	O	O	O
	DBA_SYS_PRIVS	O	O	O	O	O
	DBA_TABLES	O	O	O	O	O
	DBA_TABLESPACES	O	O	O	O	O
	DBA_TAB_COLS	O	O	O	O	O
	DBA_TAB_COLUMNS	O	O	O	O	O
	DBA_TAB_COMMENTS	O	O	O	O	O
	DBA_TAB_IDENTITY_COLS	O	O	O	O	O

所属	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	DBA_TAB_PLACE	X	O	O	O	O
	DBA_TAB_PRIVS	O	O	O	O	O
	DBA_TAB_SHARDS	X	O	O	O	O
	DBA_TBS_PRIVS	O	O	O	O	O
	DBA_USERS	O	O	O	O	O
	DBA_VIEWS	O	O	O	O	O
Views of USER_family	USER_ALL_TABLES	O	O	O	O	O
	USER_ARGUMENTS	X	O	O	O	O
	USER_CATALOG	O	O	O	O	O
	USER_CLUSTER_TABLES	X	O	O	O	O
	USER_COL_COMMENTS	O	O	O	O	O
	USER_COL_PLACE	X	X	X	X	X
	USER_COL_PRIVS	O	O	O	O	O
	USER_COL_PRIVS_MADE	O	O	O	O	O
	USER_COL_PRIVS_RECD	O	O	O	O	O
	USER_CONSTRAINTS	O	O	O	O	O
	USER_CONS_COLUMNS	O	O	O	O	O

所属	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	USER_DEPENDENCIES	X	O	O	O	O
	USER_EXTENTS	O	O	O	O	O
	USER_GLOBAL_SECONDARY_INDEXES	X	O	O	O	O
	USER_GSI_PLACE	X	O	O	O	O
	USER_INDEXES	O	O	O	O	O
	USER_IND_COLUMNS	O	O	O	O	O
	USER_IND_PLACE	X	O	O	O	O
	USER_OBJECTS	O	O	O	O	O
	USER_PACKAGE_PRIVS	X	X	O	O	O
	USER_PACKAGE_PRIVS_MADE	X	X	O	O	O
	USER_PACKAGE_PRIVS_RECD	X	X	O	O	O
	USER_PROCEDURES	X	O	O	O	O
	USER_PROC_PRIVS	X	O	O	O	O
	USER_PROC_PRIVS_MADE	X	O	O	O	O
	USER_PROC_PRIVS_RECD	X	O	O	O	O
	USER_RECYCLEBIN	X	X	O	O	O
	USER_SCHEMAS	O	O	O	O	O

所属	特征	2.x	3.x	V5.020	V5.021	V5.022
	USER_SCHEMA_PATH	O	O	O	O	O
	USER_SCHEMA_PRIVS	O	O	O	O	O
	USER_SCHEMA_PRIVS_MADE	O	O	O	O	O
	USER_SCHEMA_PRIVS_RECD	O	O	O	O	O
	USER_SEQUENCES	O	O	O	O	O
	USER_SEQ_PRIVS	O	O	O	O	O
	USER_SEQ_PRIVS_MADE	O	O	O	O	O
	USER_SEQ_PRIVS_RECD	O	O	O	O	O
	USER_SHARD_KEY_COLUMNS	X	O	O	O	O
	USER_SOURCE	X	O	O	O	O
	USER_SYNONYMS	O	O	O	O	O
	USER_SYS_PRIVS	O	O	O	O	O
	USER_TABLES	O	O	O	O	O
	USER_TABLESPACES	O	O	O	O	O
	USER_TAB_COLS	O	O	O	O	O
	USER_TAB_COLUMNS	O	O	O	O	O
	USER_TAB_COMMENTS	O	O	O	O	O

所属	特征	2.x	3.x	V5.020	V5.021	V5.022
	USER_TAB_IDENTITY_COLS	O	O	O	O	O
	USER_TAB_PLACE	X	O	O	O	O
	USER_TAB_PRIVS	O	O	O	O	O
	USER_TAB_PRIVS_MADE	O	O	O	O	O
	USER_TAB_PRIVS_RECD	O	O	O	O	O
	USER_TAB_SHARDS	X	O	O	O	O
	USER_USERS	O	O	O	O	O
	USER_VIEWS	O	O	O	O	O
Other views	AUDIT_POLICIES	X	O	O	O	O
	AUDIT_POLICY_ENABLED	X	O	O	O	O
	AUDIT_POLICY_OPTIONS	X	O	O	O	O
	AUDIT_TRAIL	X	O	O	O	O
	DATABASE_PROPERTIES	O	O	O	O	O
	DBC_TABLE_TYPE_INFO	O	O	O	O	O
	DICTIONARY	O	O	O	O	O
	DICT_COLUMNS	O	O	O	O	O
	IMPLEMENTATION_INFO	O	O	O	O	O

所属	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	IMPLEMENTATION_INFO_BASE	O	O	O	O	O
	JDBC_CLIENT_PROPS	O	O	O	O	O
	PRODUCT	O	O	O	O	O
	SESSION_PRIVS	O	O	O	O	O
	SUPPLEMENTAL_LOG_TABLE_INFO	O	O	O	O	O
Aliased synonym	COLS	O	O	O	O	O
	DICT	O	O	O	O	O
	IND	O	O	O	O	O
	OBJ	O	O	O	O	O
	RECYCLEBIN	X	X	O	O	O
	SEQ	O	O	O	O	O
	TABS	O	O	O	O	O

Table 4-5 DICTIONARY_SCHEMA 模式的特征矩阵

INFORMATION_SCHEMA模式

以下为INFORMATION_SCHEMA模式的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
COLUMNS	O	O	O	O	O
COLUMN_PRIVILEGES	O	O	O	O	O
CONSTRAINT_COLUMN_USAGE	O	O	O	O	O
CONSTRAINT_TABLE_USAGE	O	O	O	O	O
INFORMATION_SCHEMA_CATALOG_NAME	O	O	O	O	O
KEY_COLUMN_USAGE	O	O	O	O	O
MODULES	X	X	O	O	O
MODULE_BODY	X	X	O	O	O
MODULE_BODY_MODULE_USAGE	X	X	O	O	O
MODULE_BODY_ROUTINE_USAGE	X	X	O	O	O
MODULE_BODY_SEQUENCE_USAGE	X	X	O	O	O
MODULEBODY_TABLE_USAGE	X	X	O	O	O
MODULE_MODULE_USAGE	X	X	O	O	O
MODULE_PRIVILEGES	X	X	O	O	O
MODULE_ROUTINE_USAGE	X	X	O	O	O
MODULE_SEQUENCE_USAGE	X	X	O	O	O
MODULE_TABLE_USAGE	X	X	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
PARAMETERS	X	O	O	O	O
REFERENTIAL_CONSTRAINTS	O	O	O	O	O
ROUTINES	X	O	O	O	O
ROUTINE_MODULE_USAGE	X	X	O	O	O
ROUTINE_PRIVILEGES	X	O	O	O	O
ROUTINE_ROUTINE_USAGE	X	O	O	O	O
ROUTINE_SEQUENCE_USAGE	X	O	O	O	O
ROUTINE_TABLE_USAGE	X	O	O	O	O
SCHEMATA	O	O	O	O	O
SEQUENCES	O	O	O	O	O
SQL_FEATURES	O	O	O	O	O
SQL_IMPLEMENTATION_INFO	O	O	O	O	O
SQL_PACKAGES	O	O	O	O	O
SQL_PARTS	O	O	O	O	O
SQL_SIZING	O	O	O	O	O
STATISTICS	O	O	O	O	O
TABLES	O	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
TABLE_CONSTRAINTS	O	O	O	O	O
TABLE_PRIVILEGES	O	O	O	O	O
USAGE_PRIVILEGES	O	O	O	O	O
VIEWS	O	O	O	O	O
VIEW_MODULE_USAGE	X	X	O	O	O
VIEW_ROUTINE_USAGE	X	O	O	O	O
VIEW_TABLE_USAGE	O	O	O	O	O

Table 4-6 INFORMATION_SCHEMA模式的特征矩阵

PERFORMANCE_VIEW_SCHEMA模式

以下为PERFORMANCE_VIEW_SCHEMA模式的特征矩阵

特征	2.x	3.x	V5.020	V5.021	V5.022
GV$___	X	O	O	O	O
V$AGABLE_INFO	X	O	O	O	O
V$ARCHIVELOG	O	O	O	O	O
V$AUDITABLE_DB_PRIVILEGES	X	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
V$AUDITABLE_SYSTEM_ACTIONS	X	O	O	O	O
V$BACKUP	O	O	O	O	O
V$BALANCER	O	O	O	O	O
V$BCH	X	X	O	O	O
V$BUFFER_STAT	X	X	O	O	O
V$CLUSTER_DISPATCHER	X	O	O	O	O
V$CLUSTER_LOCATION	X	O	O	O	O
V$CLUSTER_MEMBER	X	O	O	O	O
V$COLUMNS	O	O	O	O	O
V$CONTROLFILE	O	O	O	O	O
V$DATAFILE	O	O	O	O	O
V$DB_CHANGE_TRACKING	X	X	O	O	O
V$DB_FILE	O	O	O	O	O
V$DB_PROPERTY	X	X	X	X	O
V$DISPATCHER	O	O	O	O	O
V$ERROR_CODE	O	O	O	O	O
V$GLOBAL_TRANSACTION	O	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
V$JOURNALING	X	O	O	O	O
V$INCREMENTAL_BACKUP	O	O	O	O	O
V$INSTANCE	O	O	O	O	O
V$KEYWORDS	O	O	O	O	O
V$LATCH	O	O	O	O	O
V$LOCK_WAIT	O	O	O	O	O
V$LOCKED_OBJECT	X	X	O	O	O
V$LOGFILE	O	O	O	O	O
V$OPEN_CURSOR	X	X	X	X	O
V$PLAN_HISTORY	X	X	X	O	O
V$PLAN_HISTORY_LATEST	X	X	X	O	O
V$PROCESS_MEM_STAT	O	O	O	O	O
V$PROCESS_SQL_STAT	O	O	O	O	O
V$PROCESS_STAT	O	O	O	O	O
V$PROPERTY	O	O	O	O	O
V$PROPERTY_ALIAS	X	X	X	X	O
V$PSM_RESERVED_WORDS	X	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
V$QUEUE	O	O	O	O	O
V$RESERVED_WORDS	O	O	O	O	O
V$SESSION	O	O	O	O	O
V$SESSION_AUDIT	X	O	O	O	O
V$SESSION_CONNECT_INFO	O	O	O	O	O
V$SESSION_EVENT	X	O	O	O	O
V$SESSION_MEM_STAT	O	O	O	O	O
V$SESSION_MEM_USAGE	X	X	X	O	O
V$SESSION_SQL_STAT	O	O	O	O	O
V$SESSION_STAT	O	O	O	O	O
V$SESSION_WAIT	X	O	O	O	O
V$SHARED_MODE	O	O	O	O	O
V$SHARED_SERVER	O	O	O	O	O
V$SHM_SEGMENT	O	O	O	O	O
V$SPROPERTY	O	O	O	O	O
V$SQLFN_METADATA	O	O	O	O	O
V$SQL_CACHE	O	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
V$SQL_COMMAND	X	O	O	O	O
V$SQL_HISTORY	X	O	O	O	O
V$STATEMENT	O	O	O	O	O
V$SYSTEM_EVENT	X	O	O	O	O
V$SYSTEM_MEM_STAT	O	O	O	O	O
V$SYSTEM_SQL_STAT	O	O	O	O	O
V$SYSTEM_STAT	O	O	O	O	O
V$TABLES	O	O	O	O	O
V$TABLESPACE	O	O	O	O	O
V$TABLESPACE_STAT	X	O	O	O	O
V$TRANSACTION	O	O	O	O	O
V$WAIT_EVENT_CLASS_NAME	X	O	O	O	O
V$WAIT_EVENT_NAME	X	O	O	O	O
V$XA_TRANSATION	X	O	O	O	O

Table 4-7 PERFORMANCE_VIEW_SCHEMA模式的特征矩阵

服务器属性(Server Property)

以下为服务器属性的特征矩阵

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
ADMIN_SESSION_POOL_INIT_SIZE	X	X	X	X	O
ADMIN_SESSION_POOL_NEXT_SIZE	X	X	X	X	O
AGING_INTERVAL	O	O	O	O	O
AGING_PLAN_INTERVAL	O	O	O	O	O
ARCHIVELOG_DIR	O	X	X	X	X
ARCHIVELOG_DIR_1 ~ DIR_10	O	O	O	O	O
ARCHIVELOG_FILE	O	O	O	O	O
ARCHIVELOG_MODE	O	O	O	O	O
BACKUP_DIR_1 ~ DIR_10	O	O	O	O	O
BLOCK_READ_COUNT	O	O	O	O	O
BROADCAST_INDEX_REBUILD_PROTOCOL	X	X	X	O	O
BROADCAST_REBALANCE_PROTOCOL	X	X	O	O	O
BUFFER_CACHE_SIZE	X	X	O	O	O
BUFFER_CHECKPOINT_LIST_COUNT	X	X	O	X	X

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
BUFFER_DIRTY_PAGE_LIMIT	X	X	X	X	O
BUFFER_FLUSH_THREADS	X	X	O	X	X
BUFFER_FLUSHING_INTERVAL	X	X	O	X	X
BUFFER_FREE_LIST_COUNT	X	X	O	O	O
BUFFER_HASH_BUCKETS	X	X	O	O	O
BUFFER_HOT_REGION_CRITERIA	X	X	O	O	O
BUFFER_HOT_REGION_PERCENT	X	X	O	O	O
BUFFER_LRU_LIST_COUNT	X	X	O	O	O
BUFFER_LRU_SCAN_PERCENT	X	X	X	X	O
BUFFER_MULTIPAGE_READ_COUNT	X	X	O	O	O
BUFFER_PREFETCH_PAGE_COUNT	X	X	X	O	O
BULK_IO_PAGE_COUNT	O	O	O	O	O
CDISPATCHER_HOT_POLICY_INTERVAL	X	O	O	O	O
CDISPATCHER_LOCKABLE_THREADS	X	X	X	X	O
CDISPATCHER_LOCKLESS_THREADS	X	X	O	O	O
CDISPATCHER_MAX_PACKET_BUFFER_SIZE	X	X	X	X	O
CDISPATCHER_SOCKET_BUFFER_SIZE	X	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
CHANGE_TRACKING	X	X	O	O	O
CHANGE_TRACKING_EXTENT_SIZE	X	X	O	O	O
CHANGE_TRACKING_FILE	X	X	O	O	O
CHAR_LENGTH_UNITS	O	O	O	O	O
CHARACTER_SET	O	O	O	O	O
CHECK_DEDICATE_CONNECTION_INTERVAL	X	O	O	O	O
CHECK_DEDICATE_SOCKET	X	X	X	X	X
CHECKPOINT_LIST_COUNT_PER_IO_GROUP	X	X	X	X	O
CLIENT_MAX_COUNT	O	O	O	O	O
CLIENT_NUMA_POLICY	X	O	O	O	O
CLOSE_PSM_CHILD_STMTS	X	O	O	O	O
CLUSTER_ASYNC_COMMIT	X	O	O	O	O
CLUSTER_ASYNC_REPLICATION	X	O	O	O	X
CLUSTER_CM_BUFFER_COUNT	X	O	O	O	X
CLUSTER_CM_BUFFER_SIZE	X	O	O	O	O
CLUSTER_CM_READ_BUFFER_SIZE	X	O	O	O	O
CLUSTER_COMMIT_SLAVE_CSERVERS	X	X	X	X	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
CLUSTER_COMMIT_STREAM_ISOLATION	X	O	O	O	O
CLUSTER_CONNECTION	X	O	O	O	O
CLUSTER_CONNECTION_TIMEOUT_SEC	X	O	O	O	O
CLUSTER_DATA_SYNC_SERVERS	X	O	O	O	O
CLUSTER_DEADLOCK_TIMEOUT	X	X	O	O	O
CLUSTER_DISPATCHER_IN_QUEUE_SIZE	X	O	O	O	O
CLUSTER_DISPATCHER_NUMA_STREAM_MAP	X	O	O	O	O
CLUSTER_DISPATCHER_OUT_QUEUE_SIZE	X	O	O	O	O
CLUSTER_GSERVER_RESPONSE_QUEUE_SIZE	X	X	X	X	O
CLUSTER_HEARTBEAT_INTERVAL	X	O	O	O	O
CLUSTER_HEARTBEAT_RETRY_COUNT	X	O	O	O	O
CLUSTER_IGNORE_INACTIVE_MEMBER	X	O	O	O	O
CLUSTER_KEEPALIVE_IDLE_TIME	X	X	X	X	O
CLUSTER_LOCKABLE_CSERVERS	X	X	X	X	O
CLUSTER_LOCKLESS_CSERVERS	X	X	X	X	O
CLUSTER_MAX_PACKET_SIZE	X	O	O	O	O
CLUSTER_MAX_PAYLOAD_SIZE	X	O	O	O	O

Feature	2.x	3.x	V5.020	V5.021	V5.022
CLUSTER_PACKET_ALLOCATION_TIMEOUT	X	O	O	O	O
CLUSTER_SESSION_HASH_BUCKETS	X	X	O	O	O
CLUSTER_SPLIT_BRAIN_RESOLUTION_POLICY	X	O	O	O	O
CLUSTER_SPLIT_BRAIN_RETRY_COUNT	X	O	O	O	O
COMMITTER_HOT_POLICY_INTERVAL	X	O	O	O	O
CONTROL_FILE_0 ~ FILE_7	O	O	O	O	O
CONTROL_FILE_COUNT	O	O	O	O	O
CONTROL_FILE_TEMP_NAME	O	O	O	O	O
COORDINATOR_COMMIT_WRITE_MODE	X	O	O	O	O
DA_CLIENT_NUMA_MODE	X	O	O	O	O
DATA_STORE_MODE	O	O	O	O	O
DATABASE_ACCESS_MODE	O	O	O	O	O
DATABASE_INSTANCE_NAME	X	O	O	O	O
DDL_AUTOCOMMIT	O	O	O	O	O
DDL_LOCK_TIMEOUT	O	O	O	O	O
DEADLOCK_PRIORITY	X	X	O	O	O
DEFAULT_GLOBAL_SECONDARY_INDEX_CREATION	X	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
DEFAULT_INDEX_LOGGING	O	O	X	X	X
DEFAULT_INDEX_PCTFREE	X	O	O	O	O
DEFAULT_INITRANS	O	O	O	O	O
DEFAULT_MAXTRANS	O	O	O	O	O
DEFAULT_PCTFREE	O	O	O	O	O
DEFAULT_PCTUSED	O	O	O	O	O
DEFAULT_REMOVAL_BACKUP_FILE	O	O	O	O	O
DEFAULT_REMOVAL_OBSOLETE_BACKUP_LIST	O	O	O	O	O
DEFAULT_SHARDING	X	O	O	O	O
DISABLE_DDL_CDC_GIVEUP	O	O	O	O	O
DISABLE_UPDATE_PK_CDC_GIVEUP	O	O	O	O	O
DISALLOWED_PROTOCOL_TARGETTYPE	X	O	O	O	O
DISALLOWED_PROTOCOL_TARGETTYPE_WITH_ALL	X	O	O	O	O
DISALLOWED_PROTOCOL_TARGETTYPE_WITH_NAME	X	O	O	O	O
DISPATCHERS	O	O	O	O	O
DISPATCHER_CM_BUFFER_SIZE	O	O	O	O	O
DISPATCHER_CM_UNIT_SIZE	O	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
DISPATCHER_CONNECTIONS	O	O	O	O	O
DISPATCHER_HOT_POLICY_INTERVAL	X	O	O	O	O
DISPATCHER_LOAD_BALANCING	X	O	O	O	O
DISPATCHER_NUMA_STREAM_MAP	X	O	O	O	O
DISPATCHER_QUEUE_SIZE	O	O	O	O	O
DISPATCHER_REQUEST_MINI_QUEUE_COUNT	X	O	O	O	O
DISPATCHER_RESPONSE_MINI_QUEUE_COUNT	X	O	O	O	O
EXECUTE_INST_HASH_TABLE_USING_AVAILABLE_MEMORY	X	X	X	O	O
FETCH_FAILOVER	X	O	O	O	O
FULL_TABLE_SCAN_CACHING_THRESHOLD	X	X	X	X	O
GLOBAL_CONNECTION_ALLOW_SESSION_DEPENDENCY	X	O	O	O	O
GLOBAL_JOURNAL_BUFFER_SIZE	X	O	O	O	O
GLOBAL_JOURNAL_BUFFER_TOTAL_MAX_SIZE	X	O	O	O	O
GLOBAL_PROPERTY_LOCK_TIMEOUT	X	O	O	O	O
GLOBAL_TRANSACTION_COMMIT_WRITE_MODE	X	O	O	O	O
GLOBAL_TRANSACTION_ISOLATION_SCOPE	X	O	O	O	O
GLOBAL_TRANSACTION_LOG_DIR	X	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
GLOBAL_TRANSACTION_LOG_FILE_SIZE	X	O	O	O	O
GMASTER_NUMA_NODE	X	O	O	O	O
GMON_AUTOSTART	X	O	O	O	O
HINT_ERROR	O	O	O	O	O
IDLE_TIMEOUT	O	O	O	O	O
IN_DOUBT_DECISION	O	O	O	O	O
IN_KEY_RANGE_ARRAY_COUNT	X	X	O	O	O
INCREMENTAL_BACKUP_SCAN_BUFFER_SIZE	X	X	O	O	O
INCREMENTAL_DATAFILE_HEADER_UPDATE_CRITERIA	X	X	X	X	O
INDEX_BUILD_PARALLEL_FACTOR	O	O	O	O	O
INDEX_MERGE_RUN_COUNT	O	O	O	O	O
INDEX_REBUILD_BLOCK_READ_COUNT	X	X	O	O	O
INDEX_SORT_RUN_SIZE	O	O	O	O	O
INDEX_TREE_MERGE_PARALLEL_FACTOR	X	O	O	O	O
INST_ALLOCATOR_COUNT	X	O	O	O	O
INST_HASH_TABLE_BUCKET_MAX_COUNT	X	X	X	O	O
INST_TABLE_BLOCK_SIZE	X	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
IPC_CHANNEL_COUNT	X	X	X	O	O
JOURNAL_TEMP_DIR	X	O	O	O	O
KEEPALIVE_IDLE_TIME	O	O	O	O	O
LOCAL_CLUSTER_MEMBER	X	O	O	O	O
LOCAL_CLUSTER_MEMBER_HOST	X	O	O	O	O
LOCAL_CLUSTER_MEMBER_PORT	X	O	O	O	O
LOCAL_JOURNAL_BUFFER_SIZE	X	O	O	O	O
LOCATION_FILE	X	O	O	O	O
LOCATOR_QUERY_TIMEOUT	X	O	O	O	O
LOCK_HASH_TABLE_SIZE	O	O	O	O	O
LOCKABLE_DISPATCHER_CM_BUFFER_COUNT	X	X	X	X	O
LOCKLESS_DISPATCHER_CM_BUFFER_COUNT	X	X	X	X	O
LOG_BLOCK_SIZE	O	O	O	O	O
LOG_BUFFER_SIZE	O	O	O	O	O
LOG_DIR	O	O	O	O	O
LOG_FILE_SIZE	O	O	O	O	O
LOG_GROUP_COUNT	O	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
LOG_MIRROR_MODE	O	O	O	O	O
LOG_MIRROR_SHARED_MEMORY_STATIC_SIZE	O	O	O	O	O
LOG_MIRROR_TIMEOUT	O	O	O	O	O
LOG_SYNC_INTERVAL	O	O	O	O	O
LOG_SYNC_INTERVAL_MSEC	X	O	O	O	O
MAX_GROUP_COUNT	X	O	O	O	O
MAX_JOURNAL_FILE_SIZE	X	O	O	O	O
MAX_NODE_COUNT	X	O	O	O	O
MAXIMUM_CONCURRENT_ACTIVITIES	O	O	O	O	O
MAXIMUM_FILE_CACHE_SIZE	X	X	X	O	O
MAXIMUM_FLANGE_COUNT	X	O	O	O	O
MAXIMUM_FLUSH_BUFFER_PAGE_COUNT	X	X	X	O	O
MAXIMUM_FLUSH_LOG_BLOCK_COUNT	O	O	O	O	O
MAXIMUM_FLUSH_PAGE_COUNT	O	O	O	O	O
MAXIMUM_INDEX_REBUILD_JOURNAL_REPLAY_COUNT	X	X	O	O	O
MAXIMUM_JOURNAL_REPLAY_COUNT	X	O	O	O	O
MAXIMUM_NAMED_CURSOR_COUNT	O	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
MAXIMUM_PACKAGE_INSTANCE_COUNT	X	X	X	O	O
MAXIMUM_SESSION_CM_BUFFER_SIZE	O	O	O	O	O
MEASURE_CLUSTER_LATENCY	X	O	O	O	O
MEDIA_RECOVERY_LOG_BUFFER_SIZE	O	X	X	X	X
MIN_SAMPLE_ROW_COUNT	X	O	O	O	O
MINIMUM_UNDO_PAGE_COUNT	O	O	O	O	O
NET_BUFFER_SIZE	O	O	O	O	O
NLS_DATE_FORMAT	O	O	O	O	O
NLS_TIME_FORMAT	O	O	O	O	O
NLS_TIME_WITH_TIME_ZONE_FORMAT	O	O	O	O	O
NLS_TIMESTAMP_FORMAT	O	O	O	O	O
NLS_TIMESTAMP_WITH_TIME_ZONE_FORMAT	O	O	O	O	O
NUMA	X	O	O	O	O
NUMA_MAP	X	O	O	O	O
OFFLINE_MEMBER_AFTER_FAILOVER	X	O	O	O	O
ONLINE_INDEX_REBUILD_JOURNAL_REPLAY_THRESHOLD	X	X	O	O	O
ONLINE_JOURNAL_REPLAY_THRESHOLD	X	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
OS_GROUP_ACCESS	X	O	O	O	O
PACKET_COMPRESSION_THRESHOLD	X	X	O	O	O
PAGE_CHECKSUM_TYPE	O	O	O	O	O
PARALLEL_IO_FACTOR	O	O	O	O	O
PARALLEL_IO_GROUP_1 ~ GROUP_16	O	O	O	O	O
PARALLEL_LOAD_FACTOR	O	O	O	O	O
PENDING_LOG_BUFFER_COUNT	O	O	O	O	O
PLAN_CACHE	O	O	O	O	O
PLAN_CACHE_SIZE	O	O	O	O	O
PLAN_HISTORY	X	X	X	O	O
PLAN_HISTORY_SIZE	X	X	X	O	O
PRIVATE_STATIC_AREA_INIT_SIZE	X	X	O	O	O
PRIVATE_STATIC_AREA_NEXT_SIZE	X	X	O	O	O
PRIVATE_STATIC_AREA_SHRINK_THRESHOLD	X	X	O	O	O
PRIVATE_STATIC_AREA_SIZE	O	O	O	O	O
PROCESS_MAX_COUNT	O	O	O	O	O
QUERY_TIMEOUT	O	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
READABLE_ARCHIVELOG_DIR_COUNT	O	O	O	O	O
READABLE_BACKUP_DIR_COUNT	O	O	O	O	O
REBALANCE_BLOCK_READ_COUNT	X	O	O	O	O
REBALANCE_SHARD_DIVISOR	X	X	X	X	O
RECOMPILE_CHECK_MINIMUM_PAGE_COUNT	O	X	X	X	X
RECOMPILE_PAGE_PERCENT	O	X	X	X	X
RECOVERY_LOG_BUFFER_SIZE	X	O	O	O	O
RECYCLEBIN	X	X	O	O	O
REDO_LOG_COMPRESSION_THRESHOLD	X	O	O	O	O
REFINE_RELATION	O	O	O	O	O
SESSION_FATAL_BEHAVIOR	O	O	O	O	O
SESSION_MEMORY_INIT_SIZE	X	O	O	O	O
SESSION_MEMORY_SHRINK_THRESHOLD	X	O	O	O	O
SESSION_POOL_INIT_SIZE	X	X	X	X	O
SESSION_POOL_NEXT_SIZE	X	X	X	X	O
SHARED_MEMORY_ADDRESS	O	O	O	O	O
SHARED_MEMORY_STATIC_KEY	O	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
SHARED_MEMORY_STATIC_NAME	O	O	O	O	O
SHARED_MEMORY_STATIC_SIZE	O	O	O	O	O
SHARED_REQUEST_QUEUE_COUNT	O	O	O	O	O
SHARED_SERVERS	O	O	O	O	O
SHARED_SESSION	O	O	O	O	O
SNAPSHOT_STATEMENT_TIMEOUT	O	O	O	O	O
SQL_HISTORY_SIZE	X	O	O	O	O
SUPPLEMENTAL_LOG_DATA_PRIMARY_KEY	O	O	O	O	O
SYNC_DISPATCHER_CM_BUFFER_COUNT	X	X	X	X	O
SYSTEM_DISK_DATA_TABLESPACE_SIZE	X	X	O	O	O
SYSTEM_FILE_IO	O	O	O	O	O
SYSTEM_MEMORY_AUX_TABLESPACE_SIZE	X	O	O	O	O
SYSTEM_MEMORY_DATA_TABLESPACE_SIZE	O	O	O	O	O
SYSTEM_MEMORY_DICT_TABLESPACE_SIZE	O	O	O	O	O
SYSTEM_MEMORY_TEMP_TABLESPACE_SIZE	O	O	O	O	O
SYSTEM_MEMORY_UNDO_TABLESPACE_SIZE	O	O	O	O	O
SYSTEM_TABLESPACE_DIR	O	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
SYSTEM_UDS_DIR	X	O	O	O	O
TCP_NODELAY	X	O	O	O	O
TEMP_SEGMENT_CACHE_SIZE	X	O	O	O	O
TEMP_UNDO_ENABLED	X	O	O	O	O
TIMED_STATISTICS	X	O	O	O	O
TIMER_INTERVAL	O	X	X	O	O
TIMEZONE	O	O	O	O	O
TRACE_ALTER_SYSTEM	O	O	O	O	O
TRACE_DDL	O	O	O	O	O
TRACE_LOG_ID	O	O	O	O	O
TRACE_LOG_MSGBUG_SIZE	X	O	O	O	O
TRACE_LOG_TIME_DETAIL	O	O	O	O	O
TRACE_LOGGER	X	O	O	O	O
TRACE_LOGGER_REMOTE_HOST	X	O	O	O	O
TRACE_LOGGER_REMOTE_PORT	X	O	O	O	O
TRACE_LOGIN	O	O	O	O	O
TRACE_LONG_RUN_CURSOR	O	O	O	O	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
TRACE_LONG_RUN_SQL	O	O	O	O	O
TRACE_LONG_RUN_TIMER	X	X	O	O	O
TRACE_SYSTEM_DIR	X	X	X	X	O
TRACE_XA	O	O	O	O	O
TRANSACTION_ALLOCATION_TIMEOUT	X	O	O	O	O
TRANSACTION_COMMIT_WRITE_MODE	O	O	O	O	O
TRANSACTION_MAXIMUM_UNDO_PAGE_COUNT	O	O	O	O	O
TRANSACTION_TABLE_SIZE	O	O	O	O	O
TRANSACTION_TIMEOUT	X	O	O	O	O
UNDO_RELATION_ALLOCATION_TIMEOUT	X	O	O	O	O
UNDO_RELATION_COUNT	O	O	O	O	O
UNDO_SHRINK_THRESHOLD	O	O	O	O	O
USE_LARGE_PAGES	X	X	O	O	O
USER_DATA_TABLESPACE_MEDIA_TYPE	X	X	O	O	O
USER_DATA_TABLESPACE_SIZE	X	X	O	O	O
USER_DISK_DATA_TABLESPACE_NEXTSIZE	X	X	O	O	O
USER_TEMP_TABLESPACE_SIZE	O	O	O	O	O

Feature	2.x	3.x	V5.020	V5.021	V5.022
XA_TRANSACTION_IDLE_TIMEOUT	X	X	O	O	O

Table 4-8 服务器属性的特征矩阵

Property Alias

以下为Property alias的特征矩阵

特征	2.x	3.x	V5.020	V5.021	V5.022
CDISPATCHER_THREADS	X	O	O	O	O
CLUSTER_COMMIT_SLAVES	X	O	O	O	O
CLUSTER_SERVER_RESPONSE_QUEUE_SIZE	X	O	O	O	O
CSERVER	X	O	O	O	O
INCREMENTAL_CHECKPOINT_CRITERIA	X	X	X	X	O
LOCKLESS_CSERVERS	X	X	O	O	O
MEMORY_MERGE_RUN_COUNT	O	O	O	O	O
MEMORY_SORT_RUN_SIZE	O	O	O	O	O
SYSTEM_LOGGER_DIR	O	O	O	O	O

Table 4-9 Property alias的特征矩阵

SQL

SQL Element

数据类型

以下为数据类型的的特征矩阵

类型	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
字符串类型	CHAR	O	O	O	O	O
	VARCHAR	O	O	O	O	O
	LONG VARCHAR	O	O	O	O	O
二进制字符串类型	BINARY	O	O	O	O	O
	VARBINARY	O	O	O	O	O
	LONG VARBINARY	O	O	O	O	O
十进制数字类型	SMALLINT	O	O	O	O	O
	INTEGER	O	O	O	O	O
	BIGINT	O	O	O	O	O
	NUMERIC	O	O	O	O	O
	DECIMAL	X	O	O	O	O

类型	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	NUMBER	O	O	O	O	O
	REAL	O	O	O	O	O
	DOUBLE PRECISION	O	O	O	O	O
	FLOAT	O	O	O	O	O
二进制数字类型	NATIVE_SMALLINT	O	O	O	O	O
	NATIVE_INTEGER	O	O	O	O	O
	NATIVE_BIGINT	O	O	O	O	O
	NATIVE_REAL	O	O	O	O	O
	NATIVE_DOUBLE	O	O	O	O	O
BOOLEAN类型	BOOLEAN	O	O	O	O	O
日期/时间类型	DATE	O	O	O	O	O
	TIME	O	O	O	O	O
	TIME WITH TIME ZONE	O	O	O	O	O
	TIMESTAMP	O	O	O	O	O
	TIMESTAMP WITH TIME ZONE	O	O	O	O	O
INTERVAL类	INTERVAL YEAR TO MONTH	O	O	O	O	O

类型	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
型	INTERVAL YEAR	O	O	O	O	O
	INTERVAL MONTH	O	O	O	O	O
	INTERVAL DAY TO SECOND	O	O	O	O	O
	INTERVAL DAY	O	O	O	O	O
	INTERVAL HOUR	O	O	O	O	O
	INTERVAL MINUTE	O	O	O	O	O
	INTERVAL SECOND	O	O	O	O	O
	INTERVAL DAY TO HOUR	O	O	O	O	O
	INTERVAL DAY TO MINUTE	O	O	O	O	O
	INTERVAL HOUR TO MINUTE	O	O	O	O	O
	INTERVAL HOUR TO SECOND	O	O	O	O	O
	INTERVAL MINUTE TO SECOND	O	O	O	O	O
ROWID类型	ROWID	O	O	O	O	O

Table 4-10 数据类型的特征矩阵

函数(Function)

以下为函数及运算符的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
expr1 * expr2	O	O	O	O	O
expr1 + expr2	O	O	O	O	O
datetime + interval	O	O	O	O	O
＋ expr	O	O	O	O	O
expr1 - expr2	O	O	O	O	O
datetime - interval	O	O	O	O	O
- expr	O	O	O	O	O
expr1 / expr2	O	O	O	O	O
str1 || str2	O	O	O	O	O
expr <comp> expr	O	O	O	O	O
expr <comp> (subquery)	O	O	O	O	O
(subquery) <comp> expr	O	O	O	O	O
(subquery) <comp> (subquery)	O	O	O	O	O
(expr, ...) <comp> (expr, ...)	O	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
(expr, ...) <comp> (subquery)	O	O	O	O	O
(subquery) <comp> (expr, ...)	O	O	O	O	O
expr <comp> {ALL|ANY|SOME} (expr, ...)	O	O	O	O	O
expr <comp> {ALL|ANY|SOME} (subquery)	O	O	O	O	O
(subquery) <comp> {ALL|ANY|SOME} (expr, ...)	O	O	O	O	O
(subquery) <comp> {ALL|ANY|SOME} (subquery)	O	O	O	O	O
(expr, ...) <comp> {ALL|ANY|SOME} (expr_list, ...)	O	O	O	O	O
(expr, ...) <comp> {ALL|ANY|SOME} (subquery)	O	O	O	O	O
(subquery) <comp> {ALL|ANY|SOME} (expr_list, ...)	O	O	O	O	O
ABS(num)	O	O	O	O	O
ACOS(num)	O	O	O	O	O
ADDDATE(date, interval)	O	O	O	O	O
ADDDATE(expr, days)	O	O	O	O	O
ADDTIME(expr1, expr2)	O	O	O	O	O
ADD_MONTHS(date, number)	O	O	O	O	O
AND	O	O	O	O	O
ASCII(char)	X	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
ASIN(num)	O	O	O	O	O
ATAN(num)	O	O	O	O	O
ATAN2(num1, num2)	O	O	O	O	O
AVG(num)	O	O	O	O	O
AVG(expr) OVER	X	X	X	X	O
expr1 [NOT] BETWEEN [ASYMMETRIC|SYMMETRIC] expr2 AND expr3	O	O	O	O	O
BITAND(num1, num2)	O	O	O	O	O
BITNOT(num)	O	O	O	O	O
BITOR(num1, num2)	O	O	O	O	O
BITXOR(num1, num2)	O	O	O	O	O
BIT_LENGTH(str)	O	O	O	O	O
BYTE_LENGTH(str)	O	O	O	O	O
CASE .. WHEN .. THEN .. ELSE .. END	O	O	O	O	O
CASE2(condition, result, ...)	O	O	O	O	O
CAST(expr AS datatype)	O	O	O	O	O
CBRT(num)	O	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
CEIL(num)	O	O	O	O	O
CEILING(num)	O	O	O	O	O
CHAR_LENGTH(str)	O	O	O	O	O
CHARACTER_LENGTH(str)	O	O	O	O	O
CHR(num)	X	O	O	O	O
CLOCK_DATE()	O	O	O	O	O
CLOCK_LOCALTIME()	O	O	O	O	O
CLOCK_LOCALTIMESTAMP()	O	O	O	O	O
CLOCK_TIME()	O	O	O	O	O
CLOCK_TIMESTAMP()	O	O	O	O	O
COALESCE(expr1, ..., exprN)	O	O	O	O	O
CONCAT(str1, str2)	O	O	O	O	O
CONCATENATE(str1, str2)	O	O	O	O	O
CONNECT_BY_ISCYCLE	X	X	X	O	O
CONNECT_BY_ISLEAF	X	X	X	O	O
CONNECT_BY_ROOT expr	X	X	X	O	O
CORR(expr1, expr2) OVER	X	X	X	X	O

特征	2.x	3.x	V5.020	V5.021	V5.022
COS(num)	O	O	O	O	O
COT(num)	O	O	O	O	O
COUNT(expr)	O	O	O	O	O
COUNT(expr) OVER	X	X	X	X	O
COUNT(*)	O	O	O	O	O
COUNT(*) OVER	X	X	X	X	O
COVAR_POP(expr1, expr2) OVER	X	X	X	X	O
COVAR_SAMP(expr1, expr2) OVER	X	X	X	X	O
CUME_DIST() OVER	X	X	X	X	O
CURRENT_CATALOG	O	O	O	O	O
CURRENT_DATE	O	O	O	O	O
CURRENT_SCHEMA	O	O	O	O	O
CURRENT_TIME	O	O	O	O	O
CURRENT_TIMESTAMP	O	O	O	O	O
CURRENT_USER	O	O	O	O	O
seq.CURRVAL	O	O	O	O	O
CURRVAL(seq)	O	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
DATEADD(datepart, number, date)	O	O	O	O	O
DATEDIFF(datepart, startdate, enddate)	O	O	O	O	O
DATE_ADD(date, interval)	O	O	O	O	O
DATE_PART(field, datetime)	O	O	O	O	O
DECODE(expr, comparison, result, ...)	O	O	O	O	O
DEGREES(radians)	O	O	O	O	O
DENSE_RANK() OVER	X	X	X	X	O
DIGEST (data, type)	X	O	O	O	O
expr IS [NOT] DISTINCT FROM expr	X	X	X	X	O
(expr, ...) IS [NOT] DISTINCT FROM (expr, ...)	X	X	X	X	O
DUMP(expr)	O	O	O	O	O
EXISTS(subquery)	O	O	O	O	O
EXP(num)	O	O	O	O	O
EXTRACT(field FROM datetime)	O	O	O	O	O
FACTORIAL(num)	O	O	O	O	O
FIRST : aggr_func KEEP (DENSE_RANK FIRST ORDER BY expr, ...) OVER	X	X	X	X	O

特征	2.x	3.x	V5.020	V5.021	V5.022
FIRST_VALUE(expr) OVER	X	X	X	X	O
FLOOR(num)	O	O	O	O	O
FROM_BASE64(str)	X	O	O	O	O
FROM_TZ(timestamp, timezone)	X	X	X	O	O
GREATEST(expr, ...)	O	O	O	O	O
HEX(str)	X	O	O	O	O
expr1 [NOT] IN (expr, ...)	O	O	O	O	O
expr1 [NOT] IN (subquery)	O	O	O	O	O
subquery [NOT] IN (<expr_list>)	O	O	O	O	O
subquery [NOT] IN (subquery)	O	O	O	O	O
<expr_list> [NOT] IN (<expr_list>, ...)	O	O	O	O	O
<expr_list> [NOT] IN (subquery)	O	O	O	O	O
subquery [NOT] IN (<expr_list>, ...)	O	O	O	O	O
INITCAP(str)	O	O	O	O	O
INSTR(str, substr, ...)	O	O	O	O	O
IS NOT NULL	O	O	O	O	O
IS NULL	O	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
LAG(expr [, offset [, default]]) OVER	X	X	X	X	O
LAST : aggr_func KEEP (DENSE_RANK LAST ORDER BY expr, ...) OVER	X	X	X	X	O
LAST_DAY(date)	O	O	O	O	O
LAST_IDENTITY_VALUE()	X	O	O	O	O
LAST_VALUE(expr) OVER	X	X	X	X	O
LEAD(expr [, offset [, default]]) OVER	X	X	X	X	O
LEAST(expr, ...)	O	O	O	O	O
LENGTH(str)	O	O	O	O	O
LENGTHB(str)	O	O	O	O	O
LEVEL	X	X	X	O	O
string [NOT] LIKE pattern ESCAPE escape_char	O	O	O	O	O
LISTAGG(str [, delimiter]) OVER	X	X	X	X	O
LN(num)	O	O	O	O	O
LNNVL(expr)	X	X	O	O	O
LOCALTIME	O	O	O	O	O
LOCALTIMESTAMP	O	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
LOCAL_GROUP_ID()	X	O	O	O	O
LOCAL_GROUP_NAME()	X	O	O	O	O
LOCAL_MEMBER_ID()	X	O	O	O	O
LOCAL_MEMBER_NAME()	X	O	O	O	O
LOG(num2)	O	O	O	O	O
LOG(num1, num2)	O	O	O	O	O
LOGON_USER()	O	O	O	O	O
LOWER(str)	O	O	O	O	O
LPAD(str, length, fill)	O	O	O	O	O
LTRIM(str, [str])	O	O	O	O	O
MAX(expr)	O	O	O	O	O
MAX(expr) OVER	X	X	X	X	O
MEDIAN(expr) OVER	X	X	X	X	O
MIN(expr)	O	O	O	O	O
MIN(expr) OVER	X	X	X	X	O
MOD(num1, num2)	O	O	O	O	O
MONTHS_BETWEEN(date1, date2)	X	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
NEXT_DAY(date, day)	X	O	O	O	O
seq.NEXTVAL	O	O	O	O	O
NEXTVAL(seq)	O	O	O	O	O
NEXT VALUE FOR seq	O	O	O	O	O
NOT	O	O	O	O	O
NTH_VALUE(expr, n) OVER	X	X	X	X	O
NTILE(expr) OVER	X	X	X	X	O
NULLIF(expr1, expr2)	O	O	O	O	O
NUMTODSINTERVAL(num, interval_indicator)	X	X	O	O	O
NUMTOYMINTERVAL(num, interval_indicator)	X	X	O	O	O
NVL(expr1, expr2)	O	O	O	O	O
NVL2(expr1, expr2, expr3)	O	O	O	O	O
OCTET_LENGTH(str)	O	O	O	O	O
OVERLAY(str1 PLACING str2 FROM start FOR length)	O	O	O	O	O
OR	O	O	O	O	O
PERCENT_RANK() OVER	X	X	X	X	O
PERCENTILE_CONT(expr) OVER	X	X	X	X	O

特征	2.x	3.x	V5.020	V5.021	V5.022
PERCENTILE_DISC(expr) OVER	X	X	X	X	O
PHYSICAL_LENGTH(expr)	X	X	O	O	O
PI()	O	O	O	O	O
POSITION(str1 IN str2)	O	O	O	O	O
POWER(num1, num2)	O	O	O	O	O
PRIOR expr	X	X	X	O	O
RADIANS(degrees)	O	O	O	O	O
RANDOM(min, max)	O	O	O	O	O
RANK() OVER	X	X	X	X	O
RATIO_TO_REPORT(expr) OVER	X	X	X	X	O
REGR_AVGX(expr1, expr2) OVER	X	X	X	X	O
REGR_AVGY(expr1, expr2) OVER	X	X	X	X	O
REGR_COUNT(expr1, expr2) OVER	X	X	X	X	O
REGR_INTERCEPT(expr1, expr2) OVER	X	X	X	X	O
REGR_R2(expr1, expr2) OVER	X	X	X	X	O
REGR_SLOPE(expr1, expr2) OVER	X	X	X	X	O
REGR_SXX(expr1, expr2) OVER	X	X	X	X	O

特征	2.x	3.x	V5.020	V5.021	V5.022
REGR_SXY(expr1, expr2) OVER	X	X	X	X	O
REGR_SYY(expr1, expr2) OVER	X	X	X	X	O
REPEAT(str, num)	O	O	O	O	O
REPLACE(str, from, to)	O	O	O	O	O
REVERSE(str)	X	O	O	O	O
ROUND(num)	O	O	O	O	O
ROUND(date, fmt)	O	O	O	O	O
ROW_NUMBER() OVER	X	X	X	X	O
ROWID_GRID_BLOCK_ID(rowid)	X	O	O	O	O
ROWID_GRID_BLOCK_SEQ(rowid)	X	O	O	O	O
ROWID_MEMBER_ID(rowid)	X	O	O	O	O
ROWID_OBJECT_ID(rowid)	O	O	O	O	O
ROWID_PAGE_ID(rowid)	O	O	O	O	O
ROWID_ROW_NUMBER(rowid)	O	O	O	O	O
ROWID_SHARD_ID(rowid)	X	O	O	O	O
ROWID_TABLESPACE_ID(rowid)	O	O	O	O	O
ROWNUM	X	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
RPAD(str, length, fill)	O	O	O	O	O
RTRIM(str, [str])	O	O	O	O	O
SESSION_ID()	O	O	O	O	O
SESSION_SERIAL()	O	O	O	O	O
SESSION_USER	O	O	O	O	O
SESSIONTIMEZONE()	X	X	X	X	O
SHARD_GROUP_ID(table, expr)	X	O	O	O	O
SHARD_GROUP_NAME(table_name, shard_key_value [, ...])	X	O	O	O	O
SHARD_ID(table, expr)	X	O	O	O	O
SHARD_NAME(table_name, shard_key_value [, ...])	X	O	O	O	O
SHIFT_LEFT(num, cnt)	O	O	O	O	O
SHIFT_RIGHT(num, cnt)	O	O	O	O	O
SIGN(num)	O	O	O	O	O
SIN(num)	O	O	O	O	O
SPLIT_PART(str, delimiter, field)	O	O	O	O	O
SQRT(num)	O	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
STATEMENT_DATE()	O	O	O	O	O
STATEMENT_LOCALTIME()	O	O	O	O	O
STATEMENT_LOCALTIMESTAMP()	O	O	O	O	O
STATEMENT_TIME()	O	O	O	O	O
STATEMENT_TIMESTAMP()	O	O	O	O	O
STATEMENT_VIEW_SCN()	O	O	O	O	O
STATEMENT_VIEW_SCN_DCN()	X	O	O	O	O
STATEMENT_VIEW_SCN_GCN()	X	O	O	O	O
STATEMENT_VIEW_SCN_LCN()	X	O	O	O	O
STDDEV([ALL | DISTINCT] expr)	X	O	O	O	O
STDDEV(expr) OVER	X	X	X	X	O
STDDEV_POP(expr)	X	O	O	O	O
STDDEV_POP(expr) OVER	X	X	X	X	O
STDDEV_SAMP(expr)	X	O	O	O	O
STDDEV_SAMP(expr) OVER	X	X	X	X	O
STRING_AGG(str [, delimiter]) OVER	X	X	X	X	O
SUBSTR(str FROM start FOR length)	O	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
SUBSTR(str, start, length)	O	O	O	O	O
SUBSTRB(str, start, length)	O	O	O	O	O
SUBSTRING(str FROM start FOR length)	O	O	O	O	O
SUBSTRING(str, start, length)	O	O	O	O	O
SUM(expr)	O	O	O	O	O
SUM(expr) OVER	X	X	X	X	O
SYSDATE	O	O	O	O	O
SYS_CONNECT_BY_PATH(expr, 'string')	X	X	X	O	O
SYS_EXTRACT_UTC(datetime_with_timezone)	X	O	O	O	O
SYSTIME	O	O	O	O	O
SYSTIMESTAMP	O	O	O	O	O
TAN(num)	O	O	O	O	O
TO_CHAR(datetime, fmt)	O	O	O	O	O
TO_CHAR(number, fmt)	O	O	O	O	O
TO_BASE64(str)	X	O	O	O	O
TO_DATE(str, fmt)	O	O	O	O	O
TO_NATIVE_BIGINT(str, fmt)	X	X	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
TO_NATIVE_DOUBLE(str, fmt)	O	O	O	O	O
TO_NATIVE_INTEGER(str, fmt)	X	X	O	O	O
TO_NATIVE_REAL(str, fmt)	O	O	O	O	O
TO_NATIVE_SMALLINT(str, fmt)	X	X	O	O	O
TO_NUMBER(num, fmt)	O	O	O	O	O
TO_TIME(str, fmt)	O	O	O	O	O
TO_TIME_TZ(str, fmt)	O	O	O	O	O
TO_TIME_WITH_TIME_ZONE(str, fmt)	O	O	O	O	O
TO_TIMESTAMP(str, fmt)	O	O	O	O	O
TO_TIMESTAMP_TZ(str, fmt)	O	O	O	O	O
TO_TIMESTAMP_WITH_TIME_ZONE(str, fmt)	O	O	O	O	O
TRANSACTION_DATE()	O	O	O	O	O
TRANSACTION_LOCALTIME()	O	O	O	O	O
TRANSACTION_LOCALTIMESTAMP()	O	O	O	O	O
TRANSACTION_TIME()	O	O	O	O	O
TRANSACTION_TIMESTAMP()	O	O	O	O	O
TRANSLATE(str, from, to)	O	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
TRIM(LEADING|TRAILING|BOTH trim_char FROM source)	O	O	O	O	O
TRUNC(num, scale)	O	O	O	O	O
TRUNC(date, fmt)	O	O	O	O	O
UPPER(str)	O	O	O	O	O
UNHEX(str)	X	O	O	O	O
UNHEX_TO_CHARSTR(str)	X	O	O	O	O
USER_ID()	O	O	O	O	O
UUID()	X	O	O	O	O
VAR_POP(expr)	X	O	O	O	O
VAR_POP(expr) OVER	X	X	X	X	O
VAR_SAMP(expr)	X	O	O	O	O
VAR_SAMP(expr) OVER	X	X	X	X	O
VARIANCE([ALL | DISTINCT] expr)	X	O	O	O	O
VARIANCE(expr) OVER	X	X	X	X	O
VERSION()	O	O	O	O	O
WIDTH_BUCKET(num, min, max, cnt)	O	O	O	O	O

Table 4-11 函数的特征矩阵

对象(Object)

SQL 对象(SQL Object)

以下为生成/删除/变更SQL对象的DDL的特征矩阵

对象	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
数据库对象	ALTER DATABASE ARCHIVELOG	O	O	O	O	O
	ALTER DATABASE ADD LOGFILE	O	O	O	O	O
	ALTER DATABASE DROP LOGFILE	O	O	O	O	O
	ALTER DATABASE RENAME GLOBAL TRANSACTION LOGFILE	X	X	X	X	O
	ALTER DATABASE RENAME LOGFILE	O	O	O	O	O
	ALTER DATABASE BEGIN/END BACKUP	O	O	O	O	O
	ALTER DATABASE RECOVER	O	O	O	O	O
	ALTER DATABASE RECOVER TABLESPACE	O	O	O	O	O
	ALTER DATABASE REGISTER	O	O	O	O	O
	ALTER DATABASE RESTORE	O	O	O	O	O
	ANALYZE SYSTEM	X	O	O	O	O

对象	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	COMMENT ON object IS ..	O	O	O	O	O
Profile对象	CREATE PROFILE	O	O	O	O	O
	DROP PROFILE	O	O	O	O	O
	ALTER PROFILE	O	O	O	O	O
审计策略对象	CREATE AUDIT POLICY	X	O	O	O	O
	DROP AUDIT POLICY	X	O	O	O	O
	ALTER AUDIT POLICY	X	O	O	O	O
	AUDIT POLICY	X	O	O	O	O
	NOAUDIT POLICY	X	O	O	O	O
授权对象	CREATE USER	O	O	O	O	O
	DROP USER	O	O	O	O	O
	ALTER USER	O	O	O	O	O
	GRANT privileges TO	O	O	O	O	O
	REVOKE privileges FROM	O	O	O	O	O
模式对象	CREATE SCHEMA	O	O	O	O	O
	DROP SCHEMA	O	O	O	O	O
表空间对	CREATE MEMORY DATA TABLESPACE	O	O	O	O	O

对象	特征	2.x	3.x	V5.020	V5.021	V5.022
象	CREATE MEMORY TEMPORARY TABLESPACE	O	O	O	O	O
	DROP TABLESPACE	O	O	O	O	O
	ALTER TABLESPACE .. RENAME TO	O	O	O	O	O
	ALTER TABLESPACE .. BEGIN/END BACKUP	O	O	O	O	O
	ALTER TABLESPACE .. ADD [DATAFILE|MEMORY]	O	O	O	O	O
	ALTER TABLESPACE .. DROP [DATAFILE|MEMORY]	O	O	O	O	O
	ALTER TABLESPACE .. RENAME DATAFILE	O	O	O	O	O
	ALTER TABLESPACE .. {ONLINE|OFFLINE}	O	O	O	O	O
表对象	CREATE TABLE	O	O	O	O	O
	CREATE TABLE AS SELECT	O	O	O	O	O
	CREATE GLOBAL TEMPORARY TABLE	X	O	O	O	O
	CREATE GLOBAL TEMPORARY TABLE AS SELECT	X	O	O	O	O
	CREATE IMMUTABLE TABLE	X	X	O	O	O
	CREATE IMMUTABLE TABLE AS SELECT	X	X	O	O	O
	DROP TABLE	O	O	O	O	O

对象	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	TRUNCATE TABLE	O	O	O	O	O
	ALTER TABLE .. STORAGE	O	O	O	O	O
	ALTER TABLE .. RENAME TO	O	O	O	O	O
	ALTER TABLE .. ADD COLUMN	O	O	O	O	O
	ALTER TABLE .. SET UNUSED COLUMN	O	O	O	O	O
	ALTER TABLE .. ALTER COLUMN	O	O	O	O	O
	ALTER TABLE .. RENAME COLUMN	O	O	O	O	O
	ALTER TABLE .. RENAME CONSTRAINT	X	O	O	O	O
	ALTER TABLE .. ADD CONSTRAINT	O	O	O	O	O
	ALTER TABLE .. DROP CONSTRAINT	O	O	O	O	O
	ALTER TABLE .. ALTER CONSTRAINT	O	O	O	O	O
	ALTER TABLE .. ADD SUPPLEMENTAL LOG	O	O	O	O	O
	ALTER TABLE .. DROP SUPPLEMENTAL LOG	O	O	O	O	O
	ALTER TABLE .. READ { ONLY | WRITE }	X	O	O	O	O
	ANALYZE TABLE	X	O	O	O	O
	FLASHBACK TABLE	X	X	O	O	O
	PURGE	X	X	O	O	O

对象	特征	2.x	3.x	V5.020	V5.021	V5.022
视图对象	CREATE VIEW	O	O	O	O	O
	DROP VIEW	O	O	O	O	O
	ALTER VIEW	O	O	O	O	O
索引对象	CREATE INDEX	O	O	O	O	O
	DROP INDEX	O	O	O	O	O
	ALTER INDEX .. AGING	X	O	O	O	O
	ALTER INDEX .. STORAGE	O	O	O	O	O
	ALTER INDEX .. RENAME	X	O	O	O	O
	ALTER INDEX .. REBUILD	X	X	O	O	O
	ALTER INDEX .. COALESCE	X	X	X	X	O
序列对象	CREATE SEQUENCE	O	O	O	O	O
	DROP SEQUENCE	O	O	O	O	O
	ALTER SEQUENCE	O	O	O	O	O
Synonym 对象	CREATE SYNONYM	O	O	O	O	O
	DROP SYNONYM	O	O	O	O	O
	CREATE PUBLIC SYNONYM	O	O	O	O	O
	DROP PUBLIC SYNONYM	O	O	O	O	O

对象	特征	2.x	3.x	V5.020	V5.021	V5.022
Stored procedure 对象	CREATE PROCEDURE	X	O	O	O	O
	DROP PROCEDURE	X	O	O	O	O
	ALTER PROCEDURE	X	O	O	O	O
Stored function 对象	CREATE FUNCTION	X	O	O	O	O
	DROP FUNCTION	X	O	O	O	O
	ALTER FUNCTION	X	O	O	O	O
Package 对象	CREATE PACKAGE	X	X	O	O	O
	CREATE PACKAGE BODY	X	X	O	O	O
	ALTER PACKAGE	X	X	O	O	O
	DROP PACKAGE	X	X	O	O	O

Table 4-12 SQL对象DDL的特征矩阵

集群对象(Cluster Object)

以下为生成/删除/变更集群对象的DDL的特征矩阵

对象	特征	2.x	3.x	V5.020	V5.021	V5.022
集群系统对象	ALTER DATABASE REBALANCE	X	O	O	O	O
	ALTER DATABASE DROP INACTIVE CLUSTER MEMBERS	X	O	O	O	O
	ALTER DATABASE DROP OFFLINE SEGMENTS	X	X	X	X	O
	ALTER DATABASE SYNCHRONIZE	X	X	X	X	O
集群群组对象	CREATE CLUSTER GROUP	X	O	O	O	O
	DROP CLUSTER GROUP	X	O	O	O	O
集群成员	ALTER CLUSTER GROUP name ADD MEMBER	X	O	O	O	O
	ALTER CLUSTER GROUP name OFFLINE MEMBER	X	O	O	O	O
	ALTER DATABASE RESET LOCAL CLUSTER MEMBER	X	O	O	O	O

对象	特征	2.x	3.x	V5.020	V5.021	V5.022
对象	ALTER SYSTEM IRRECOVERABLE CLUSTER MEMBER	X	O	O	O	O
	ALTER SYSTEM JOIN DATABASE	X	O	O	O	O
集群位置对象	CREATE CLUSTER LOCATION	X	O	O	O	O
	DROP CLUSTER LOCATION	X	O	O	O	O
	ALTER CLUSTER LOCATION	X	O	O	O	O
集群表及分片对象	ALTER TABLE name REBALANCE	X	O	O	O	O
	ALTER TABLE name DROP OFFLINE SEGMENTS	X	X	X	X	O
	ALTER TABLE name SYNCHRONIZE	X	X	X	X	O
	ALTER TABLE name MERGE SHARDS	X	X	O	O	O
	ALTER TABLE name MOVE SHARD	X	O	O	O	O
	ALTER TABLE name SPLIT SHARD	X	O	O	O	O
	ALTER TABLE name RENAME SHARD	X	O	O	O	O
全局	ALTER TABLE name ADD GLOBAL SECONDARY INDEX	X	O	O	O	O

对象	特征	2.x	3.x	V5.020	V5.021	V5.022
二级索引对象	ALTER TABLE name DROP GLOBAL SECONDARY INDEX	X	O	O	O	O
	ALTER TABLE name ALTER GLOBAL SECONDARY INDEX	X	O	O	O	O
	ALTER TABLE name ALTER GLOBAL SECONDARY INDEX REBUILD	X	X	O	O	O
	ALTER TABLE name ALTER GLOBAL SECONDARY INDEX COALESCE	X	X	X	X	O

Table 4-13 集群对象DDL的特征矩阵

SQL语言(SQL Language)

DML

以下为操作数据的DML语句的特征矩阵

特征	2.x	3.x	V5.020	V5.021	V5.022
INSERT INTO ..	O	O	O	O	O
INSERT INTO .. RETURNING query	O	O	O	O	O
INSERT INTO .. RETURNING .. INTO ..	O	O	O	O	O
INSERT INTO .. UPDATE	X	X	X	O	O
INSERT INTO .. UPDATE .. RETURNING ..	X	X	X	O	O
INSERT INTO .. UPDATE .. RETURNING .. INTO ..	X	X	X	O	O
DELETE FROM ..	O	O	O	O	O
DELETE FROM .. RETURNING query	O	O	O	O	O
DELETE FROM .. RETURNING .. INTO ..	O	O	O	O	O
DELETE FROM .. WHERE CURRENT OF cursor	O	O	O	O	O
UPDATE ..	O	O	O	O	O
UPDATE .. RETURNING query	O	O	O	O	O
UPDATE .. RETURNING .. INTO ..	O	O	O	O	O

特征	2.x	3.x	V5.020	V5.021	V5.022
UPDATE .. WHERE CURRENT OF cursor	O	O	O	O	O
CALL proc_name	X	O	O	O	O

Table 4-14 DML的特征矩阵

查询(Query）

以下为查询数据的SELECT语句的特征矩阵

特征	2.x	3.x	V5.020	V5.021	V5.022
<query expression>	O	O	O	O	O
<query specification>	O	O	O	O	O
<select list>	O	O	O	O	O
<from clause>	O	O	O	O	O
<joined table>	O	O	O	O	O
<where clause>	O	O	O	O	O
<group by clause>	O	O	O	O	O
<window clause>	X	X	X	X	O
<window partition clause>	X	X	X	X	O

特征	2.x	3.x	V5.020	V5.021	V5.022
<window order clause>	X	X	X	X	O
<window frame clause>	X	X	X	X	O
<window frame exclusion>	X	X	X	X	O
<order by clause>	O	O	O	O	O
<offset limit clause>	O	O	O	O	O
<set operator>	O	O	O	O	O
<subquery>	O	O	O	O	O
<hint clause>	O	O	O	O	O
<with clause>	X	X	X	O	O
<search clause>	X	X	X	O	O
<cycle clause>	X	X	X	O	O
<start with clause>	X	X	X	O	O
<connect by clause>	X	X	X	O	O
<order siblings by clause>	X	X	X	O	O

Table 4-15 SELECT 的特征矩阵

控制语言(Control Language)

以下为控制语句的特征矩阵

区分	特征	2.x	3.x	V5.020	V5.021	V5.022
事务	COMMIT	O	O	O	O	O
	ROLLBACK	O	O	O	O	O
	SAVEPOINT	O	O	O	O	O
	RELEASE SAVEPOINT	O	O	O	O	O
	LOCK TABLE	O	O	O	O	O
	SET CONSTRAINTS	O	O	O	O	O
	SET TRANSACTION	O	O	O	O	O
会话	SET SESSION CHARACTERISTICS AS	O	O	O	O	O
	SET SESSION AUTHORIZATION	O	O	O	O	O
	SET SCHEMA	X	X	X	O	O
	SET TIME ZONE	O	O	O	O	O
	ALTER SESSION SET property	O	O	O	O	O
系统	ALTER SYSTEM {OPEN|MOUNT} DATABASE	O	O	O	O	O
	ALTER SYSTEM CHECKPOINT	O	O	O	O	O

区分	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	ALTER SYSTEM KILL SESSION	O	O	O	O	O
	ALTER SYSTEM RECONNECT GLOBAL CONNECTION	X	O	O	O	O
	ALTER SYSTEM SWITCH LOGFILE	O	O	O	O	O
	ALTER SYSTEM SET property	O	O	O	O	O
	ALTER SYSTEM RESET property	O	O	O	O	O

Table 4-16 控制语句的特征矩阵

PSM语言(PSM Language)

以下为Persistent Stored Module（PSM）语言元素的特征矩阵

特征	2.x	3.x	V5.020	V5.021	V5.022
Assignment Statement	X	O	O	O	O
Basic LOOP Statement	X	O	O	O	O
Block (BEGIN .. END)	X	O	O	O	O
CASE Statement	X	O	O	O	O
CLOSE Statement	X	O	O	O	O
Collection Method Invocation	X	O	O	O	O
Collection Variable Declaration	X	O	O	O	O
CONTINUE Statement	X	O	O	O	O
Cursor FOR LOOP Statement	X	O	O	O	O
Cursor Variable Declaration	X	O	O	O	O
DELETE Statement Extension	X	O	O	O	O
EXCEPTION_INIT Pragma	X	O	O	O	O
Exception Declaration	X	O	O	O	O
Exception Handler	X	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
EXECUTE IMMEDIATE Statement	X	O	O	O	O
EXIT Statement	X	O	O	O	O
Explicit Cursor Declaration and Definition	X	O	O	O	O
FETCH Statement	X	O	O	O	O
FOR LOOP Statement	X	O	O	O	O
GOTO Statement	X	O	O	O	O
IF Statement	X	O	O	O	O
Implicit Cursor Attribute	X	O	O	O	O
INSERT Statement Extension	X	O	O	O	O
Named Cursor Attribute	X	O	O	O	O
NULL Statement	X	O	O	O	O
OPEN Statement	X	O	O	O	O
OPEN FOR Statement	X	O	O	O	O
Procedure Call	X	O	O	O	O
Procedure Declaration and Definition	X	O	O	O	O
RAISE Statement	X	O	O	O	O
Record Variable Declaration	X	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
RETURN Statement	X	O	O	O	O
RETURN TABLE Statement	X	X	X	X	O
RETURNING INTO clause	X	O	O	O	O
%ROWTYPE Attribute	X	O	O	O	O
Scalar Variable Declaration	X	O	O	O	O
SELECT INTO Statement	X	O	O	O	O
SQLCODE Function	X	O	O	O	O
SQLERRM Function	X	O	O	O	O
%TYPE Attribute	X	O	O	O	O
UPDATE Statement Extension	X	O	O	O	O
WHILE LOOP Statement	X	O	O	O	O

Table 4-17 Persistent Stored Module (PSM) language element的特征矩阵

API

ODBC

以下为标准ODBC API的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
SQLAllocHandle()	O	O	O	O	O
SQLBindCol()	O	O	O	O	O
SQLBindParameter()	O	O	O	O	O
SQLCloseCursor()	O	O	O	O	O
SQLColAttribute()	O	O	O	O	O
SQLColumnPrivileges()	O	O	O	O	O
SQLColumns()	O	O	O	O	O
SQLConnect()	O	O	O	O	O
SQLDescribeCol()	O	O	O	O	O
SQLDescribeParam()	O	O	O	O	O
SQLDisconnect()	O	O	O	O	O
SQLDriverConnect()	O	O	O	O	O
SQLEndTran()	O	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
SQLExecDirect()	0	0	0	0	0
SQLExecute()	0	0	0	0	0
SQLExtendedFetch()	0	0	0	0	0
SQLFetch()	0	0	0	0	0
SQLFetchScroll()	0	0	0	0	0
SQLForeignKeys()	0	0	0	0	0
SQLFreeHandle()	0	0	0	0	0
SQLFreeStmt()	0	0	0	0	0
SQLGetConnectAttr()	0	0	0	0	0
SQLGetCursorName()	0	0	0	0	0
SQLGetData()	0	0	0	0	0
SQLGetDescField()	0	0	0	0	0
SQLGetDescRec()	0	0	0	0	0
SQLGetDiagField()	0	0	0	0	0
SQLGetDiagRec()	0	0	0	0	0
SQLGetEnvAttr()	0	0	0	0	0
SQLGetFunctions()	0	0	0	0	0
SQLGetInfo()	0	0	0	0	0

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
SQLGetStmtAttr()	0	0	0	0	0
SQLGetTypeInfo()	0	0	0	0	0
SQLMoreResults()	0	0	0	0	0
SQLNumParams()	0	0	0	0	0
SQLNumResultCols()	0	0	0	0	0
SQLParamData()	0	0	0	0	0
SQLPrepare()	0	0	0	0	0
SQLPrimaryKeys()	0	0	0	0	0
SQLProcedureColumns()	0	0	0	0	0
SQLProcedures()	0	0	0	0	0
SQLPutData()	0	0	0	0	0
SQLRowCount()	0	0	0	0	0
SQLSetConnectAttr()	0	0	0	0	0
SQLSetCursorName()	0	0	0	0	0
SQLSetDescField()	0	0	0	0	0
SQLSetDescRec()	0	0	0	0	0
SQLSetEnvAttr()	0	0	0	0	0
SQLSetPos()	0	0	0	0	0

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
SQLSetStmtAttr()	0	0	0	0	0
SQLSpecialColumns()	0	0	0	0	0
SQLStatistics()	0	0	0	0	0
SQLTablePrivileges()	0	0	0	0	0
SQLTables()	0	0	0	0	0

Table 4-18 标准ODBC特征矩阵

以下为除ODBC标准接口以外其他支持接口的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
xa_open	0	0	0	0	0
xa_close	0	0	0	0	0
xa_start	0	0	0	0	0
xa_end	0	0	0	0	0
xa_rollback	0	0	0	0	0
xa_prepare	0	0	0	0	0
xa_commit	0	0	0	0	0
xa_recover	0	0	0	0	0
xa_forget	0	0	0	0	0

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
SQLGetXaSwitch	O	O	O	O	O
SQLGetXaConnectionHandle	O	O	O	O	O
SQLGetGroupCount	X	O	O	O	O
SQLGetGroupIDs	X	O	O	O	O
SQLGetGroupName	X	O	O	O	O
SQLGetSuitableGroupID	X	O	O	O	O

Table 4-19 除ODBC标准以外的函数的特征矩阵

JDBC

以下为JDBC的类别特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
CallableStatement	X	O	O	O	O
CommonDataSource	O	O	O	O	O
Connection	O	O	O	O	O
ConnectionPoolDataSource	O	O	O	O	O
DatabaseMetaData	O	O	O	O	O
DataSource	O	O	O	O	O
Driver	O	O	O	O	O
ParameterMetaData	O	O	O	O	O
PooledConnection	O	O	O	O	O
PreparedStatement	O	O	O	O	O
ResultSet	O	O	O	O	O
ResultSetMetaData	O	O	O	O	O
RowId	O	O	O	O	O
Savepoint	O	O	O	O	O
Statement	O	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
XAConnection	0	0	0	0	0
XADataSource	0	0	0	0	0
XAResource	0	0	0	0	0
SundbInterval	0	0	0	0	0
SundbTypes	0	0	0	0	0

Table 4-20 JDBC的类别特征矩阵

嵌入式SQL

预编译器选项(Precompiler Option)

以下为预编译器选项的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--help	O	O	O	O	O
--include-path	O	O	O	O	O
--no-prompt	O	O	O	O	O
--output	O	O	O	O	O
--unsafe-null	O	O	O	O	O
--version	O	O	O	O	O
--no-lineinfo	X	O	O	O	O
--char_map	X	O	O	O	O
--cumulative	X	X	O	O	O
--autocommit	X	X	X	O	O

Table 4-21 预编译器选项的特征矩阵

嵌入式 SQL专用语句

以下为仅可用于嵌入式SQL的SQL语句的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
EXEC SQL AT	O	O	O	O	O
EXEC SQL ATOMIC INSERT	O	O	O	O	O
EXEC SQL AUTOCOMMIT	O	O	O	O	O
EXEC SQL BEGIN DECLARE SECTION	O	O	O	O	O
EXEC SQL COMMIT RELEASE	O	O	O	O	O
EXEC SQL CONNECT	O	O	O	O	O
EXEC SQL CONTEXT ALLOCATE	O	O	O	O	O
EXEC SQL CONTEXT FREE	O	O	O	O	O
EXEC SQL CONTEXT USE	O	O	O	O	O
EXEC SQL DISCONNECT	O	O	O	O	O
EXEC SQL END DECLARE SECTION	O	O	O	O	O
EXEC SQL FOR	O	O	O	O	O
EXEC SQL INCLUDE	O	O	O	O	O
EXEC SQL INCLUDE SQLCA	O	O	O	O	O
EXEC SQL OPTION	O	O	O	O	O
EXEC SQL ROLLBACK RELEASE	O	O	O	O	O
EXEC SQL WHENEVER	O	O	O	O	O

Table 4-22 嵌入式SQL专用语句的特征矩阵

主机变量数据类型(Host Variable Data Type)

以下为可在host变量中使用的嵌入式 SQL 数据类型的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
C native type	O	O	O	O	O
struct, union	O	O	O	O	O
typedef	O	O	O	O	O
VARCHAR	O	O	O	O	O
LONG VARCHAR	O	O	O	O	O
BINARY	O	O	O	O	O
LONG VARBINARY	O	O	O	O	O
BOOLEAN	O	O	O	O	O
NUMBER	O	O	O	O	O
DATE	O	O	O	O	O
TIME	O	O	O	O	O
TIME WITH TIMEZONE	O	O	O	O	O
TIMESTAMP	O	O	O	O	O
TIMESTAMP WITH TIMEZONE	O	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
INTERVAL YEAR	O	O	O	O	O
INTERVAL MONTH	O	O	O	O	O
INTERVAL DAY	O	O	O	O	O
INTERVAL HOUR	O	O	O	O	O
INTERVAL MINUTE	O	O	O	O	O
INTERVAL SECOND	O	O	O	O	O
INTERVAL YEAR TO MONTH	O	O	O	O	O
INTERVAL DAY TO HOUR	O	O	O	O	O
INTERVAL DAY TO MINUTE	O	O	O	O	O
INTERVAL DAY TO SECOND	O	O	O	O	O
INTERVAL HOUR TO MINUTE	O	O	O	O	O
INTERVAL HOUR TO SECOND	O	O	O	O	O
INTERVAL MINUTE TO SECOND	O	O	O	O	O

Table 4-23 Host数据类型的特征矩阵

动态SQL(Dynamic SQL)

以下为动态SQL的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
SELECT .. INTO	O	O	O	O	O
EXECUTE IMMEDIATE sql	O	O	O	O	O
PREPARE stmt	O	O	O	O	O
EXECUTE stmt	O	O	O	O	O
DECLARE cursor FOR sql	O	O	O	O	O
DECLARE cursor FOR stmt	O	O	O	O	O
OPEN cursor	O	O	O	O	O
OPEN cursor USING	O	O	O	O	O
FETCH cursor INTO	O	O	O	O	O
CLOSE cursor	O	O	O	O	O
DELETE .. WHERE CURRENT OF cursor	O	O	O	O	O
UPDATE .. WHERE CURRENT OF cursor	O	O	O	O	O

Table 4-24 动态SQL的特征矩阵

PyDBC

Module

以下为PyDBC提供的pysundb的method特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
connect	X	O	O	O	O
Data	X	O	O	O	O
Time	X	O	O	O	O
Timestamp	X	O	O	O	O
DateFromTicks	X	O	O	O	O
TimeFromTicks	X	O	O	O	O
TimestampFromTicks	X	O	O	O	O
Binary	X	O	O	O	O
STRING	X	O	O	O	O
BINARY	X	O	O	O	O
NUMBER	X	O	O	O	O
DATETIME	X	O	O	O	O
ROWID	X	O	O	O	O
getDecimalSeparator	X	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
setDecimalSeparator	X	O	O	O	O

Table 4-25 pysundb method的特征矩阵

以下为pysundb module的attribute特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
apilevel	X	O	O	O	O
threadsafety	X	O	O	O	O
paramstyle	X	O	O	O	O
version	X	O	O	O	O
lowercase	X	O	O	O	O

Table 4-26 pysundb attribute的特征矩阵

Connection

以下为Connection对象的method特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
cursor	X	O	O	O	O
commit	X	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
rollback	X	O	O	O	O
close	X	O	O	O	O
getinfo	X	O	O	O	O
execute	X	O	O	O	O
set_attr	X	O	O	O	O

Table 4-27 Connection method的特征矩阵

以下为Connection对象的attribute特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
autocommit	X	O	O	O	O
searchescape	X	O	O	O	O
timeout	X	O	O	O	O

Table 4-28 Connection attribute的特征矩阵

Cursor

以下为Cursor对象的method特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
excute	X	O	O	O	O
executemany	X	O	O	O	O
fetchone	X	O	O	O	O
fetchall	X	O	O	O	O
fetchmany	X	O	O	O	O
commit	X	O	O	O	O
rollback	X	O	O	O	O
skip	X	O	O	O	O
nextset	X	O	O	O	O
close	X	O	O	O	O
setinputsizes	X	O	O	O	O
setoutputsize	X	O	O	O	O
callproc	X	O	O	O	O
callfunc	X	O	O	O	O
tables	X	O	O	O	O
columns	X	O	O	O	O
statistics	X	O	O	O	O
rowldColumns	X	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
rowVerColumns	X	O	O	O	O
primaryKeys	X	O	O	O	O
foreignKeys	X	O	O	O	O
procedures	X	O	O	O	O
getTypeInfo	X	O	O	O	O

Table 4-29 Cursor method的特征矩阵

以下为Cursor对象的attribute特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
Description	X	O	O	O	O
rowcount	X	O	O	O	O
arraysize	X	O	O	O	O
connection	X	O	O	O	O
fast_executemany	X	O	O	O	O

Table 4-30 Cursor attribute的特征矩阵

Row

以下为Row对象的attribute特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
cursor_description	X	O	O	O	O

Table 4-31 Row attribute的特征矩阵

工具(Utility)

gcreatedb

命令用法(Command Usage)

以下为gcreatedb命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--character_set	O	O	O	O	O
--char_length_units	O	O	O	O	O
--cluster	X	O	O	O	O
--db_comment	O	O	O	O	O
--help	O	O	O	O	O
--host	X	O	O	O	O
--member	X	O	O	O	O
--port	X	O	O	O	O
--silent	O	O	O	O	O
--timezone	O	O	O	O	O

Table 4-32 gcreatedb命令用法的特征矩阵

glsnr

命令用法(Command Usage)

以下为glsnr命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--help	O	O	O	O	O
--home	X	O	O	O	O
--silent	O	O	O	O	O
--start	O	O	O	O	O
--status	O	O	O	O	O
--stop	O	O	O	O	O

Table 4-33 glsnr命令用法的特征矩阵

配置文件(Configuration file)

以下为配置glsnr的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
BACKLOG	O	O	O	O	O
DEFAULT_CS_MODE	O	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
LISTENER_LOG_DIR	X	O	O	O	O
LISTEN_PORT	O	O	O	O	O
TCP_EXCLUDED	O	O	O	O	O
TCP_INVITED	O	O	O	O	O
TCP_HOST	O	O	O	O	O
TCP_VALIDNODE_CHECKING	O	O	O	O	O
TIMEOUT	O	O	O	O	O
USR_DIR	X	O	O	O	O

Table 4-34 glsnr配置文件syntax的特征矩阵

gsql/gsqlnet

命令用法(Command Usage)

以下为gsql命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
username password	O	O	O	O	O
--as {SYSDBA|ADMIN}	O	O	O	O	O
--conn-string	O	O	O	O	O
--dsn	O	O	O	O	O
--enable-color	O	O	O	O	O
--help	O	O	O	O	O
--import	O	O	O	O	O
--no-prompt	O	O	O	O	O
--prompt	O	O	O	O	O
--silent	O	O	O	O	O
--version	O	O	O	O	O

Table 4-35 gsql命令用法特征矩阵

交互式gsql命令(interactive gsql command)

以下为gsql对话窗口中使用的交互式gsql命令的特征矩阵

特征	2.x	3.x	V5.020	V5.021	V5.022
\\	O	O	O	O	O
\connect userid password [as sysdba]	O	O	O	O	O
\cshutdown	X	O	O	O	O
\cstartup	X	O	O	O	O
\ddl_cluster	X	O	O	O	O
\ddl_db	O	O	O	O	O
\ddl_tablespace	O	O	O	O	O
\ddl_profile	O	O	O	O	O
\ddl_audit_policy	X	O	O	O	O
\ddl_auth	O	O	O	O	O
\ddl_schema	O	O	O	O	O
\ddl_publicsynonym	O	O	O	O	O
\ddl_table	O	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
\ddl_constraint	O	O	O	O	O
\ddl_index	O	O	O	O	O
\ddl_view	O	O	O	O	O
\ddl_sequence	O	O	O	O	O
\ddl_synonym	O	O	O	O	O
\ddl_procedure	X	O	O	O	O
\ddl_package	X	X	O	O	O
\desc	O	O	O	O	O
\dynamic sql :var	O	O	O	O	O
\exec	O	O	O	O	O
\exec :var := :value	O	O	O	O	O
\exec sql	O	O	O	O	O
\explain plan [on|only]	O	O	O	O	O
\help	O	O	O	O	O
\history	O	O	O	O	O
\host {os_command}	X	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
\import	O	O	O	O	O
\idesc	O	O	O	O	O
\{n}	O	O	O	O	O
\prepare sql	O	O	O	O	O
\print	O	O	O	O	O
\quit	O	O	O	O	O
\set autocommit	O	O	O	O	O
\set color	O	O	O	O	O
\set colsize	O	O	O	O	O
\set ddlsize	O	O	O	O	O
\set error	O	O	O	O	O
\set history	O	O	O	O	O
\set linesize	O	O	O	O	O
\set numsize	O	O	O	O	O
\set pagesize	O	O	O	O	O
\set sqlprompt	X	X	X	X	O

特征	2.x	3.x	V5.020	V5.021	V5.022
\set timing	O	O	O	O	O
\set vertical	O	O	O	O	O
\shutdown {abort|immediate|transactional|normal}	O	O	O	O	O
\startup {nomount|mount|open}	O	O	O	O	O
\var	O	O	O	O	O

Table 4-36 交互式gsql命令的特征矩阵

gloader/gloadernet

命令用法(Command Usage)

以下为gloader命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
username password	O	O	O	O	O
--array	O	O	O	O	O
--atomic	O	O	O	O	O
--bad	O	O	O	O	O
--buffered	O	O	O	O	O
--commit	O	O	O	O	O
--control	O	O	O	O	O
--data	O	O	O	O	O
--dsn	O	O	O	O	O
--errors	O	O	O	O	O
--export	O	O	O	O	O
--fieldterm	X	O	O	O	O
--filesize	O	O	O	O	O
--format	O	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--help	O	O	O	O	O
--import	O	O	O	O	O
--lineterm	X	O	O	O	O
--log	O	O	O	O	O
--no-prompt	O	O	O	O	O
--parallel	O	O	O	O	O
--propagation	O	O	O	O	O
--qualifier	X	O	O	O	O
--silent	O	O	O	O	O
--AsTIMESTAMP	O	O	O	O	O
--where	X	O	O	O	O
--group-id	X	O	O	O	O
--directio-size	X	O	O	O	O

Table 4-37 gloader 命令用法的特征矩阵

控制文件语法(Control file syntax)

以下为gloader控制文件语法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
CHARACTERSET	O	O	O	O	O
FIELDS TERMINATED BY	O	O	O	O	O
OPTIONALLY ENCLOSED BY	O	O	O	O	O
TABLE table_name	O	O	O	O	O
TABLE schema_name.table_name	O	O	O	O	O
LTRIM	X	O	O	O	O
RTRIM	X	O	O	O	O
LINES TERMINATED BY	X	O	O	O	O
WHERE	X	O	O	O	O

Table 4-38 gloader控制文件语法的特征矩阵

gdump

命令用法(Command Usage)

以下为gdump命令用法的特征矩阵如下

特征		2.x	3.x	V5.020	V5.021	V5.022
Common arguments	--silent	O	O	O	O	O
File type	BACKUP	O	O	O	O	O
	COMMIT_LOG	X	O	O	O	O
	CONTROL	O	O	O	O	O
	DATA	O	O	O	O	O
	LOG	O	O	O	O	O
	LOG_BUFFER	X	O	O	O	O
	PEND_BUFFER	X	O	O	O	O
	PROPERTY	O	O	O	O	O
BACKUP file arguments	--body	O	O	O	O	O
	--tbs	O	O	O	O	O
	--number	O	O	O	O	O
	--fetch	O	O	O	O	O

特征		2.x	3.x	V5.0 20	V5.0 21	V5.0 22
CONTROL file arguments	--section	O	O	O	O	O
DATA file arguments	--header	O	O	O	O	O
	--number	O	O	O	O	O
	--fetch	O	O	O	O	O
LOG file arguments	--all	X	O	O	O	O
	--fetch	O	O	O	O	O
	--header	X	O	O	O	O
	--number	O	O	O	O	O
	--offset	O	O	O	O	O

Table 4-39 gdump命令用法的特征矩阵

tablediff

配置文件

以下为tablediff配置文件的特征矩阵

特征		2.x	3.x	V5.0 20	V5.0 21	V5.0 22
Source table	SOURCE_PASSWORD	O	O	O	O	O
	SOURCE_SCHEMA	O	O	O	O	O
	SOURCE_TABLE	O	O	O	O	O
	SOURCE_URL	O	O	O	O	O
	SOURCE_USER	O	O	O	O	O
Target table	TARGET_PASSWORD	O	O	O	O	O
	TARGET_SCHEMA	O	O	O	O	O
	TARGET_TABLE	O	O	O	O	O
	TARGET_URL	O	O	O	O	O
	TARGET_USER	O	O	O	O	O
Sync operation	TARGET_INSERT	O	O	O	O	O
	TARGET_UPDATE	O	O	O	O	O
	TARGET_DELETE	O	O	O	O	O

特征		2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	SOURCE_INSERT	O	O	O	O	O
Operation options	DIFF_BIN_FILE	O	O	O	O	O
	DIFF_OUT_FILE	O	O	O	O	O
	DISPLAY_CALL_STACK	O	O	O	O	O
	DISPLAY_ROW_UNIT	O	O	O	O	O
	EXCLUDE_COLUMNS	O	O	O	O	O
	LOGGING_ON_DIFF	O	O	O	O	O
	LOGGING_ON_SUCCESS	O	O	O	O	O
	JOB_QUEUE_SIZE	O	O	O	O	O
	JOB_THREAD	O	O	O	O	O
	JOB_UNIT_SIZE	O	O	O	O	O
	PARTITION_RANGE	O	O	O	O	O
	SYNC_OUT_FILE	O	O	O	O	O
	WHERE_CLAUSE	O	O	O	O	O

Table 4-40 tablediff配置文件的特征矩阵

gsyncher

命令用法(Command Usage)

以下为gsyncher命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--log	O	O	O	O	O
--silent	O	O	O	O	O
--home	X	O	O	O	O
--copy-right	O	O	O	O	O
--backup-path	O	O	O	O	O
--help	O	O	O	O	O

Table 4-41 gsyncher命令用法的特征矩阵

gmon

命令用法(Command Usage)

以下为gmon命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--start	X	O	O	O	O
--stop	X	O	O	O	O
--status	X	O	O	O	O
--home	X	O	O	O	O
--uds_dir	X	X	O	O	O
--silent	X	O	O	O	O
--no-copyright	X	O	O	O	O
--help	X	O	O	O	O

Table 4-42 gmon命令用法的特征矩阵

gtrclogger

命令用法(Command Usage)

以下为gtrclogger命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--dir	X	O	O	O	O
--help	X	O	O	O	O
--port	X	O	O	O	O
--start	X	O	O	O	O
--stop	X	O	O	O	O

Table 4-43 gtrclogger命令用法的特征矩阵

glocator

命令用法(Command Usage)

以下为glocator命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--create	X	O	O	O	O
--start	X	O	O	O	O
--stop	X	O	O	O	O
--conf	X	O	O	O	O
--status	X	O	O	O	O
--sync	X	O	O	O	O
--silent	X	O	O	O	O
--no-copyright	X	O	O	O	O
--help	X	O	O	O	O

Table 4-44 glocator命令用法的特征矩阵

配置文件

以下为glocator配置文件的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
PORT	X	O	O	O	O
WORKER_COUNT	X	O	O	O	O
SESSION_QUEUE_SIZE	X	O	O	O	O
SESSION_ALLOCATOR_SIZE	X	O	O	O	O
PACKET_ALLOCATOR_SIZE	X	O	O	O	O
SYSTEM_LOGGER_DIR	X	O	O	O	O
SYSTEM_UDS_DIR	X	O	O	O	O
LOCATION_FILE_DIR	X	O	O	O	O
LOCATION_FILE_SIZE	X	O	O	O	O
LOCATION_FILE_MAX_SIZE	X	O	O	O	O
SESSION_TIMEOUT	X	O	O	O	O
FAILOVER_TIMEOUT	X	O	O	O	O
ALTERNATE_LOCATORS	X	O	O	O	O
SYNC_RETRY_COUNT	X	O	O	O	O
SYNC_RESPONSE_TIMEOUT	X	O	O	O	O

Table 4-45 glocator配置文件的特征矩阵

gagent

命令用法(Command Usage)

gagent命令用法的特征矩阵如下

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--start	X	O	O	O	O
--stop	X	O	O	O	O
--conf	X	O	O	O	O
--status	X	O	O	O	O
--home	X	O	O	O	O
--silent	X	O	O	O	O
--no-copyright	X	O	O	O	O
--help	X	O	O	O	O

Table 4-46 gagent命令用法的特征矩阵

配置文件

以下为gagent配置文件的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
PORT	X	O	O	O	O
LOCATOR_HOST	X	O	O	O	O
LOCATOR_PORT	X	O	O	O	O
COMMAND_QUEUE_SIZE	X	O	O	O	O
COMMAND_ALLOCATOR_SIZE	X	O	O	O	O
PACKET_ALLOCATOR_SIZE	X	O	O	O	O
SYSTEM_LOGGER_DIR	X	O	O	O	O
SESSION_TIMEOUT	X	O	O	O	O
UPDATE_LOCATION_TIME	X	O	O	O	O
ALTERNATE_LOCATORS	X	O	O	O	O

Table 4-47 gagent配置文件的特征矩阵

gloctl

命令用法(Command Usage)

以下为gloctl命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--dsn	X	X	X	X	X
--conf	X	O	O	O	O
--ip	X	O	O	O	O
--port	X	O	O	O	O
--import	X	O	O	O	O
--silent	X	O	O	O	O
--no-copyright	X	O	O	O	O
--help	X	O	O	O	O

Table 4-48 gloctl命令用法的特征矩阵

配置文件

以下为gloctl配置文件的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
PORT	X	O	O	O	O
LOCATOR_HOST	X	O	O	O	O
LOCATOR_PORT	X	O	O	O	O

Table 4-49 gloctl配置文件的特征矩阵

Replication

cyclone

命令用法(Command Usage)

以下为cyclone命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--conf	O	O	O	O	O
--encrypt	X	O	O	O	O
--group	O	O	O	O	O
--help	O	O	O	O	O
--key	X	O	O	O	O
--master	O	O	O	O	O
--reset	O	O	O	O	O
--silent	O	O	O	O	O
--slave	O	O	O	O	O
--start	O	O	O	O	O
--status	O	O	O	O	O
--stop	O	O	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--sync	O	O	O	O	O
--stand-alone	X	O	O	O	O
--recovery	X	O	O	O	O
--local	X	O	O	O	O

Table 4-50 cyclone命令用法的特征矩阵

配置文件

以下为cyclone配置文件的特征矩阵

配置	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
Common configuration	COMM_CHUNK_COUNT	O	O	O	O	O
	DSN	O	O	O	O	O
	USER_ENCRYPT_PW	X	O	O	O	O
	GROUP_NAME	O	O	O	O	O
	HOST_IP	O	O	O	O	O
	HOST_EXTERNAL_IP	X	O	O	O	O
	HOST_PORT	O	O	O	O	O
	PORT	O	O	O	O	O

配置	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	PROTOCOL	X	O	O	O	O
	USER_ID	O	O	O	O	O
	USER_PW	O	O	O	O	O
MASTER configuration	CAPTURE_TABLE	O	O	O	O	O
	LOG_PATH	O	O	O	O	O
	READ_LOG_BLOCK_COUNT	O	O	O	O	O
	TRANS_SORT_AREA_SIZE	O	O	O	O	O
	TRANS_FILE_PATH	O	O	O	O	O
	SYNCHER_COUNT	O	O	O	O	O
	SYNC_ARRAY_SIZE	O	O	O	O	O
	GIVEUP_INTERVAL	O	O	O	O	O
	LOG_CAPTURE_INTERVAL_1	X	O	O	O	O
	LOG_CAPTURE_INTERVAL_2	X	O	O	O	O
SLAVE configuration	APPLIER_COUNT	O	O	O	O	O
	APPLY_ARRAY_SIZE	O	X	X	X	X
	APPLY_COMMIT_SIZE	O	O	O	O	O
	APPLY_TABLE	O	O	O	O	O

配置	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	MASTER_IP	O	O	O	O	O
	PROPAGATE_MODE	O	O	O	O	O
	CLUSTER	X	O	O	O	O
	ORACLE_DRIVER	X	O	O	O	O

Table 4-51 cyclone配置文件的特征矩阵

clustone

命令用法(Command Usage)

以下为clustone命令用法的特征矩阵

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--conf	X	X	X	X	O
--encrypt	X	X	X	X	O
--group	X	X	X	X	O
--help	X	X	X	X	O
--key	X	X	X	X	O
--master	X	X	X	X	O
--reset	X	X	X	X	O
--silent	X	X	X	X	O
--slave	X	X	X	X	O
--start	X	X	X	X	O
--status	X	X	X	X	O
--stop	X	X	X	X	O
--stand-alone	X	X	X	X	O
--dump	X	X	X	X	O

Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--info	X	X	X	X	O

Table 4-52 clustone命令用法的特征矩阵

配置文件

以下为clustone配置文件的特征矩阵

Configuration	Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
Common configuration	COMM_CHUNK_COUNT	X	X	X	X	O
	DSN	X	X	X	X	O
	USER_ENCRYPT_PW	X	X	X	X	O
	GROUP_NAME	X	X	X	X	O
	HOST_IP	X	X	X	X	O
	HOST_EXTERNAL_IP	X	X	X	X	O
	HOST_PORT	X	X	X	X	O
	PORT	X	X	X	X	O
	PROTOCOL	X	X	X	X	O
	USER_ID	X	X	X	X	O
	USER_PW	X	X	X	X	O

Configuration	Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	HEARTBEAT_TIMEOUT	X	X	X	X	O
MASTER configuration	CAPTURE_TABLE	X	X	X	X	O
	TXDATA_FILE_SIZE	X	X	X	X	O
	TXDATA_FILE_PATH	X	X	X	X	O
	LOG_PATH	X	X	X	X	O
	READ_LOG_BLOCK_COUNT	X	X	X	X	O
	TRANS_SORT_AREA_SIZE	X	X	X	X	O
	TRANS_FILE_PATH	X	X	X	X	O
	GIVEUP_INTERVAL	X	X	X	X	O
	LOG_CAPTURE_INTERVAL_1	X	X	X	X	O
	LOG_CAPTURE_INTERVAL_2	X	X	X	X	O
	PACKET_COMPRESSION_MODE	X	X	X	X	O
SLAVE configuration	APPLIER_COUNT	X	X	X	X	O
	APPLY_COMMIT_SIZE	X	X	X	X	O
	APPLY_TABLE	X	X	X	X	O
	MASTER_IP	X	X	X	X	O
	PROPAGATE_MODE	X	X	X	X	O

Configuration	Feature	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
	CLUSTER	X	X	X	X	O

Table 4-53 clustone配置文件的特征矩阵

logmirror

命令用法(Command Usage)

以下为logmirror命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--conf	O	O	O	O	O
--help	O	O	O	O	O
--infiniband	O	O	O	O	O
--master	O	O	O	O	O
--silent	O	O	O	O	O
--slave	O	O	O	O	O
--start	O	O	O	O	O
--stop	O	O	O	O	O

Table 4-54 logmirror命令用法的特征矩阵

配置文件

以下为logmirror配置文件的特征矩阵

配置	特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
Common configuration	PORT	O	O	O	O	O
MASTER configuration	DSN	O	O	O	O	O
	HOST_IP	O	O	O	O	O
	HOST_PORT	O	O	O	O	O
	PROTOCOL	X	O	O	O	O
	USER_ID	O	O	O	O	O
	USER_PW	O	O	O	O	O
SLAVE configuration	LOG_PATH	O	O	O	O	O
	MASTER_IP	O	O	O	O	O

Table 4-55 logmirror配置文件的特征矩阵

cymon

命令用法(Command Usage)

以下为cymon命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--conf	O	O	O	O	O
--help	O	O	O	O	O
--cycle	O	O	O	O	O
--key	X	O	O	O	O
--start	O	O	O	O	O
--stop	O	O	O	O	O
--status	O	O	O	O	O

Table 4-56 cymon命令用法的特征矩阵

cyfile

命令用法(Command Usage)

以下为cyfile命令用法的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
--conf	X	X	O	O	O
--help	X	X	O	O	O
--reset	X	X	O	O	O
--key	X	X	O	O	O
--silent	X	X	O	O	O
--info	X	X	O	O	O
--start	X	X	O	O	O
--stop	X	X	O	O	O
--group	X	X	O	O	O
--encrypt	X	X	O	O	O
--status	X	X	O	O	O

Table 4-57 cyfile命令用法的特征矩阵

配置文件

以下为cyfile配置文件的特征矩阵

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
DSN	X	X	O	O	O
HOST_IP	X	X	O	O	O
HOST_PORT	X	X	O	O	O
PROTOCOL	X	X	O	O	O
USER_ID	X	X	O	O	O
USER_PW	X	X	O	O	O
GROUP_NAME	X	X	O	O	O
USER_ENCRYPT_PW	X	X	O	O	O
CAPTURE_TABLE	X	X	O	O	O
READ_LOG_BLOCK_COUNT	X	X	O	O	O
TRANS_SORT_AREA_SIZE	X	X	O	O	O
TRANS_FILE_PATH	X	X	O	O	O
LOG_CAPTURE_INTERVAL_1	X	X	O	O	O
LOG_CAPTURE_INTERVAL_2	X	X	O	O	O
DATA_FILE_PATH	X	X	O	O	O

特征	2.x	3.x	V5.0 20	V5.0 21	V5.0 22
DATA_FILE_PREFIX	X	X	O	O	O
DATA_FILE_SIZE	X	X	O	O	O
UPDATE_BEFORE_VALUE	X	X	O	O	O

Table 4-58 cyfile配置文件的特征矩阵

4.2 SUNDB V5.0 22的新功能

本章简要介绍SUNDB V5.0 22新添加的功能

架构

系统架构

无变动内容

内部存储(Storage Internal)

无变动内容

事务控制（Transaction Control）

无变动内容

Backup & Recovery

无变动内容

数据库信息

DICTIONARY_SCHEMA

无变动内容

INFORMATION_SCHEMA

无变动内容

PERFORMANCE_VIEW_SCHEMA

添加了**V$OPEN_CURSOR**

添加了**V$PROPERTY_ALIAS**

添加了**V$DB_PROPERTY**

Server Property

添加了**REBALANCE_SHARD_DIVISOR**

添加了**SESSION_POOL_INIT_SIZE**

添加了**SESSION_POOL_NEXT_SIZE**

添加了**ADMIN_SESSION_POOL_INIT_SIZE**

添加了**ADMIN_SESSION_POOL_NEXT_SIZE**

DEFAULT_INDEX_PCTFREE的默认值更改为10

DEFAULT_MAXTRANS的默认值更改为32

设置系统增量备份执行标准的现有参数INCREMENTAL_CHECKPOINT_CRITERIA的名称更改为**BUFFER_DIRTY_PAGE_LIMIT**默认值也更改为0

为了改善磁盘表空间的缓冲区管理算法添加了**BUFFER_LRU_SCAN_PERCENT**参数

在集群环境中指定通信缓冲区数量的CLUSTER_CM_BUFFER_COUNT property被deprecate 代替添加了指定用于lockable, lockless, synchronization dispatcher的通信缓冲区数量的**LOCKABLE_DISPATCHER_CM_BUFFER_COUNTLOCKLESS_DISPATCHER_CM_BUFFER_COUNT SYNC_DISPATCHER_CM_BUFFER_COUNT**参数

在全部扫描磁盘表空间中创建的表时是否缓存到buffer取决于表大小的threshold为设置此threshold值增加了**FULL_TABLE_SCAN_CACHING_THRESHOLD**参数

为设置Hash instant table的预期bucket count的最大值增加了**INST_HASH_TABLE_BUCKET_MAX_COUNT**参数

SQL

SQL Element

数据类型

无变动内容

Function

添加了**DISTINCT Condition**

添加了**SESSIONTIMEZONE**

添加了**Window Function**

对象

SQL Object

无变动内容

Cluster Object

添加了**ALTER DATABASE DROP OFFLINE SEGMENTS**

添加了**ALTER DATABASE SYNCHRONIZE**

添加了**ALTER TABLE name DROP OFFLINE SEGMENTS**

添加了**ALTER TABLE name SYNCHRONIZE**

在**ALTER DATABASE MOVE SHARDALTER DATABASE REBALANCEALTER DATABASE REBALANCE EXCLUDE CLUSTER GROUPALTER TABLE name MOVE SHARDALTER TABLE name REBALANCEALTER TABLE name REBALANCE EXCLUDE CLUSTER GROUP cluster_group_list**中添加了SHARD DIVISOR和PARALLEL选项

ALTER TABLE REBUILD GLOBAL SECONDARY INDEX语句更改为**ALTER TABLE name ALTER GLOBAL SECONDARY INDEX REBUILD**

SQL语言(SQL Language)

DML

无变动内容

Query

Lateral Inline View

在**from clause**中添加了lateral inline view

Table Function Derived Table

在**from clause**中添加了table function derived table

WINDOW子句

添加了定义window function执行范围的WINDOW子句

详细内容请参考**window clause**

Control Language

无变动内容

PSM Language

Table Function

在**return clause**中添加了TABLE（table function column list）语句

可在function DDL 时定义table type创建table function

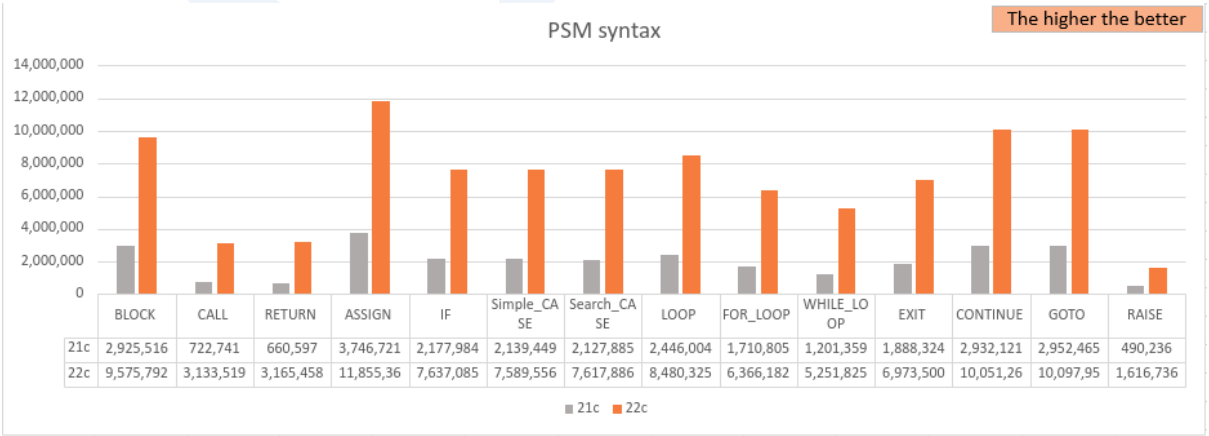
详细内容请参考**CREATE FUNCTION**

RETURN TABLE Statement

在PSM statement中添加了**RETURN TABLE Statement**

PSM Statement性能优化

如下图所示改善了PSM statement的性能



	BLOCK	CALL	RETURN	ASSIGN	IF	Simple_CASE	Search_CASE	LOOP	FOR_LOOP	WHILE_LOOP	EXIT	CONTINUE	GOTO	RAISE
21c	2,925,516	722,741	660,597	3,746,721	2,177,984	2,139,449	2,127,885	2,446,004	1,710,805	1,201,359	1,888,324	2,932,121	2,952,465	490,236
22c	9,575,792	3,133,519	3,165,458	11,855,36	7,637,085	7,589,556	7,617,886	8,480,325	6,366,182	5,251,825	6,973,500	10,051,26	10,097,95	1,616,736

API

ODBC

在**数据源构成**中添加了TRACE_POLICY

JDBC

将time和timestamp类型输出为字符串时不再丢失微秒单位

支持auto-generated key

添加了SundbTypes.REF_CURSOR

添加了JDBC非标准函数SundbPreparedStatement.setFixedCHAR()

嵌入式 SQL

gpec中支持**声明函数参数**

PDO

无变动内容

PyDBC

无变动内容

Ruby

无变动内容

Hibernate

无变动内容

Utility

gcreatedb

无变动内容

glsnr

支持IPv6

gsql/gsqlnet

添加了\set sqlprompt命令

gloader/gloadernet

无变动内容

gdump

无变动内容

tablediff

无变动内容

gsyncher

无变动内容

gmon

无变动内容

gtrclogger

无变动内容

glocator

无变动内容

gagent

无变动内容

gloctl

无变动内容

Replication

cyclone

无变动内容

clustone

无变动内容

logmirror

无变动内容

cymon

无变动内容

cyfile

无变动内容

4.3 补丁说明(Patch Notes)

V5.0 22.3 Patch Notes

ISSUE-5511 [CYCLONE] Distributor中存在internal transaction ID设置错误的情况

概要

确保在原始database中同时执行的transaction之间的transaction ID值不能重复但是完成执行的transaction ID可以在以后被再次使用

在为提取原始transaction并将其双重化而发送到slave时由于parallel apply该transaction ID的执行时间可能与原始不一样在这种情况下为了防止因transaction ID的再次使用而导致ID重复slave将独立的transaction ID更改为可以在内部区分的internal ID值

这样的internal transaction ID由distributor分配在分析特定日志的过程中使用的不是internal值而是原始值导致了一个transaction有两个transaction ID的问题发生

在Distributor中两个作为并发控制的分隔符值的transaction ID值被分配给一个transaction在特定情况下可能会发生self dead-lock

现象及症状

```
CREATE TABLE T1 ( C1 INTEGER PRIMARY KEY, C2 LONG VARCHAR );
```

```
INSERT INTO T1 VALUES( 1, 'AAA' );

DELETE FROM T1 WHERE C1=1;

INSERT INTO T1 VALUES( 1,'AAAAAAA .......' );    超过8K的数据 INSERT

COMMIT;
```

如上所示如果在一个transaction中执行处理相同key值的query和record大小超过8K的INSERT就会发生dead-lock

修改前应对

无

ISSUE-5499 Window function的参数为scalar subquery expression时无法判断值

概要

在Cluster环境中如果将可常数化的scalar subquery expression作为window function的参数使用则当执行查询时无法判断表达式结果为NULL

现象及症状

```
--# BUGBUG

--# result : 1

SELECT SUM( ( SELECT 1 FROM dual ) ) OVER() AS C_NAME

  FROM dual@G2;
```

```
C_NAME
------
  null
```

在Cluster环境中在driver node中判断所有可常数化的expression并将结果值传达给generated query

使用Subquery expression作为window function的参数时无法判断是否配置常数化因此没有实现常数化 如果必须通过Generated query传达subquery expression的结果值就会出现结果错误

虽然是Standalone或cluster环境但是只在local node执行的查询即使不进行常数化window function内的expression也会被判断所以不会发生问题

修改前应对

无

ISSUE-5505 如果Join最左边的表的access method是unique index access并且只有group by的key column中的一部分属于该unique index时则可能会产生错误的结果

概要

如果Join最左边的表的access method是unique index access并且只有group by的key column中的一部分属于该unique index时则可能会产生错误的结果

现象及症状

```
DROP TABLE IF EXISTS r;

DROP TABLE IF EXISTS s;

DROP TABLE IF EXISTS t;


CREATE TABLE r( c1 INTEGER, c2 INTEGER, c3 INTEGER, c4 INTEGER, c5

INTEGER );


INSERT INTO r VALUES(1, 1, 1, 1, 1);

INSERT INTO r VALUES(1, 2, 1, 1, 1);

INSERT INTO r VALUES(1, 2, 1, 1, 1);

INSERT INTO r VALUES(2, 1, 2, 1, 1);

INSERT INTO r VALUES(2, 1, 2, 2, 1);

INSERT INTO r VALUES(3, 1, 2, 2, 1);

INSERT INTO r VALUES(3, 2, 3, 2, 1);

INSERT INTO r VALUES(4, 1, 3, 3, 1);

INSERT INTO r VALUES(5, 1, 3, 3, 1);

INSERT INTO r VALUES(1, 3, 2, 1, 1);

INSERT INTO r VALUES(1, 1, 1, 1, 1);


COMMIT;


CREATE TABLE s( c1 INTEGER PRIMARY KEY, c2 INTEGER, c3 INTEGER, c4

INTEGER, c5 INTEGER );
```

```
INSERT INTO s VALUES(1, 1, 1, 1, 1);

INSERT INTO s VALUES(2, 3, 1, 1, 1);

INSERT INTO s VALUES(3, 3, 1, 1, 1);

INSERT INTO s VALUES(4, 2, 2, 1, 1);

INSERT INTO s VALUES(5, 2, 2, 2, 1);

INSERT INTO s VALUES(6, 1, 2, 2, 1);

INSERT INTO s VALUES(7, 1, 3, 2, 1);

INSERT INTO s VALUES(8, 3, 3, 3, 1);

INSERT INTO s VALUES(9, 3, 3, 3, 1);


COMMIT;



CREATE TABLE t( c1 INTEGER, c2 INTEGER, c3 INTEGER, c4 INTEGER, c5

INTEGER );


INSERT INTO t VALUES(1, 1, 1, 1, 1);

INSERT INTO t VALUES(1, 3, 3, 1, 1);

INSERT INTO t VALUES(1, 3, 3, 1, 1);

INSERT INTO t VALUES(1, 2, 2, 1, 1);

INSERT INTO t VALUES(1, 2, 2, 1, 1);

INSERT INTO t VALUES(2, 3, 1, 1, 1);

INSERT INTO t VALUES(3, 3, 1, 1, 1);

INSERT INTO t VALUES(4, 2, 2, 1, 1);
```

```
INSERT INTO t VALUES(5, 2, 2, 2, 1);


COMMIT;


--# result : 6 rows
\EXPLAIN PLAN
SELECT s.c1, s.c2, t.c3
  FROM s, r, t
 WHERE s.c1 > 0 AND s.c1 < 5
   AND s.c1 = r.c1
   AND r.c1 = t.c1
 GROUP BY s.c1, s.c2, t.c3;



C1 C2 C3
-- -- --
 1  1  2
 1  1  3
 1  1  1
 1  1  2
 1  1  3
 1  1  1
 1  1  2
...
18 rows selected
```

```
< Execution Plan >
=============================================================================
|  IDX  |  NODE DESCRIPTION                                                  |
-----------------------------------------------------------------------------
|    0  |  SELECT STATEMENT                                                  |
|    1  |    QUERY BLOCK ("$QB_IDX_2")                                       |
|    2  |      GROUP                                                         |
|    3  |        HASH JOIN (INNER JOIN)                                      |
|    4  |          MERGE JOIN (INNER JOIN)                                   |
|    5  |            INDEX ACCESS ("S", "S_PRIMARY_KEY_INDEX")               |
|    6  |            INDEX ACCESS ("R", "R_IDX")                             |
|    7  |          HASH JOIN INSTANT                                         |
|    8  |            TABLE ACCESS ("T")                                      |
=============================================================================
```

上述查询的执行结果应为6rows但输出了18rows 因为最左边的表s使用“S_PRIMARY_KEY_INDEX”所以join的结果对s.c1进行排列后出现 从上述plan来看group by利用join的排列结果进行grouping 但是由于那些排列的record不是对所有group key列进行排列的因此可能会出现错误的结果

修改前应对

使用/*+ USE_GROUP_HASH */ hint

ISSUE-5500 添加了JDBC非标准函数SundbPreparedStatement.setFixedCHAR()

概要

添加了JDBC非标准函数SundbPreparedStatement.setFixedCHAR()

现象及症状

SELECT查询WHERE子句中将CHAR column绑定为PreparedStatement.setSrting()时无法搜索结果

```
create table x (c char(4));
insert into x (c) values ('a');  -- inserts 'a   '
```

```
PreparedStatement stmt =
  conn.prepareStatement("select * from x where c = ?");
stmt.setString(1, "a");    // This won't return any records
stmt.executeQuery();
```

修改前应对

将CHAR column更改为VARCHAR column

ISSUE-5482 不完全恢复时若日志不足恢复可能会失败

概要

在执行不完全恢复时可以完成恢复但有时也会失败因此进行了修改

现象及症状

当Redo log丢失后使用先前备份的redo log执行不完全恢复时如果备份的redo log早于archive log则找不到archive log之后的日志所以发生问题

```
gSQL> ALTER DATABASE RECOVER UNTIL TIME '2023-04-05 19:05:01.559752';


ERR-HY000(14068): logfile does not exist -
'/sundb/sundb_data/archive_log/archive_4.log'
```

修改前应对

在没有Archive log之后的redo log的情况下如下只使用未损坏日志执行进行恢复的interactive不完全恢复

```
gSQL> ALTER DATABASE BEGIN INCOMPLETE RECOVERY;


ERR-01000(14104): Warning: suggestion '/sundb/archive_log/archive_0.log'
ERR-01000(14103): Warning: media recovery needs a logfile including log
(Lsn 139992)
Database altered.
```

```
gSQL> ALTER DATABASE RECOVER AUTOMATICALLY;

ERR-01000(14104): Warning: suggestion '/sundb/archive_log/archive_4.log'
ERR-01000(14103): Warning: media recovery needs a logfile including log
(Lsn 144143)
Database altered.

gSQL> ALTER DATABASE END INCOMPLETE RECOVERY;

Database altered.
```

ISSUE-5445 执行ADD LOGFILE GROUP时即使充分设置LOGFILE GROUP的大小也有无法添加LOGFILE GROUP的情况

概要

即使设置了足够大小的LOGFILE GROUP也无法在logfile小于最小大小的错误消息出现同时添加LOGFILE GROUP

执行ADD LOGFILE GROUP时必须以startup过程中修改的log buffer和pending log buffer个数为准计算logfile的最小大小但实际上是以property值为准计算最小大小

现象及症状

如下设置PROPERTY的状态下startup到mount阶段后如果想添加logfile group就会失败

```
LOG_BUFFER_SIZE=1G

PENDING_LOG_BUFFER_COUNT=32
```

```
gSQL> STARTUP MOUNT


Startup success


gSQL> ALTER DATABASE ADD LOGFILE GROUP 4 ('redo_4_0.log') SIZE 512M;


ERR-42000(16198): size of log file is smaller than minimum size of log

file(1107296256 bytes).
```

修改前应对

通过改变LOG_BUFFER_SIZE和PENDING_LOG_BUFFER_COUNT property来生成logfile group

ISSUE-5424 如果在Embedded SQL中没有INTO子句的情况下执行SELECT语句即使有数据也会发生错误，因此对其进行了修改

概要

在Embedded SQL中反复执行无INTO子句的SELECT语句会报错因此对其进行了修改

现象及症状

```
EXEC SQL SELECT 1 FROM DUAL;


EXEC SQL SELECT 1 FROM DUAL;
```

如上所示反复执行相同的SELECT语句时会报如下错误

```
Invalid cursor state : A cursor was open on the StatementHandle.
```

修改前应对

在SELECT语句添加INTO子句后执行

ISSUE-5411 对使用JDBC的ResultSet class的getBinaryStream方法则发生NullPointerException的问题进行了修改

概要

对获取Long varbinary类型的null数据时 若使用ResultSet class的getBinaryStream方法则发生NullPointerException的问题进行了修改

现象及症状

表TEST_LOB上LONG VARBINARY类型C_BLOB具有NULL数据

```
CREATE TABLE PUBLIC.TEST_LOB
(
```

```
   ID NUMBER( 10, 0 ),
   C_BLOB LONG VARBINARY
);

INSERT INTO TEST_LOB VALUES(1,UNHEX(HEX('XXXXXXXX')));
INSERT INTO TEST_LOB VALUES(2,null);
COMMIT;
```

以下是为获取LONG VARBINARY类型的数据使用getBinaryStream方法的代码

```
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("SELECT ID, C_BLOB FROM TEST_LOB");

while (rs.next()) {
    try{
        InputStream is = rs.getBinaryStream("c_blob");
        byte[] bytes = new byte[0];
        bytes = new byte[is.available()];
        is.read(bytes);
    } catch( Exception e){
        System.out.println(e);
    }
}
```

若执行上述代码则发生java.lang.NullPointerException

修改前应对

无

ISSUE-5398 gloader以text模式上传数据时存在数据丢失的问题对此进行了修改

概要

gloader在以text模式上传数据时使用以相同字符开头的field分隔符和line分隔符 如果数据中也包含该分隔符的第一个字符则存在数据被截断并以无效形式上传的问题对此进行了修改

现象及症状

以下为data file的示例

```
data 1^^^Cc__Cc^data 2^Rr__Rr
```

以下为control file的示例

```
TABLE TEST
FIELD TERMINATED BY '^Cc__Cc^'
LINE TERMINATED BY '^Rr__Rr\n'
```

使用上述control file和data file上传时数据会被截断

```
gloader test test -i --control test.ctl --data test.dat --array 1 --no-
copyright
```

```
COMPLETED IN IMPORTING TABLE: PUBLIC.TEST, TOTAL 3 RECORDS, SUCCEEDED 3
RECORDS


gSQL> SELECT * FROM TEST;
C1      C2
------- -------------
data 1^ null
^       null
null    data 2^Rr__Rr
```

修改前应对

使用不以相同字符开头的field分隔符和line分隔符

ISSUE-5407 等待解锁的cluster peer无法识别driver node死亡

概要

存在当远程等待解锁的driver member异常终止时远程创建的会话依然存活的问题

现象及症状

以下示例为cluster group G1拥有成员G1N1, G1N2并创建表T1的环境

在G1N1中对表T1的record进行update

```
gSQL> UPDATE T1 SET A = A + 1 WHERE A = 1;
```

```
1 row updated.
```

若对G1N2中相同record执行update则会等待

```
gSQL> UPDATE T1 SET A = 10 WHERE A = 1;
```

若查看G1N1的会话则可以查看为从G1N2发送的update而等待的cluster session

```
gSQL> SELECT SESSION_STATUS FROM V$SESSION@G1N1
       WHERE PROGRAM_NAME = 'cluster peer';


SESSION_STATUS
--------------
CONNECTED
```

即使为从G1N2发送的update而等待的gsql被杀死如下所示在G1N1中该session仍然处于存活状态

```
gSQL> SELECT SESSION_STATUS FROM V$SESSION@G1N1
       WHERE PROGRAM_NAME = 'cluster peer';


SESSION_STATUS
--------------
CONNECTED
```

修改前应对

无

V5.0 22.2 Patch Notes

ISSUE-5174 gpec中支持声明函数参数

概要

gpec中支持声明函数参数详细内容参考声明函数参数

现象及症状

无

修改前应对

无

ISSUE-5353 对Windows ODBC连接及解除时存在程序handle数量增加的问题进行了修改

概要

对使用Windows ODBC反复连接和解除时程序的整体handle数量增加的问题进行了修改

现象及症状

使用Windows ODBC反复连接和解除时程序的整体handle数量会增加

修改前应对

无

ISSUE-5344 执行View projection pruning时删除父块中使用的column的问题

概要

如果满足以下条件服务器可能会因执行错误的view projection pruning而异常终止

- 视图内部有group by或order by
- 想要从View的select list中删除的expr是不同的function expression的argument

现象及症状

以下是可能会发生问题的查询的示例

```
SELECT sum_col1
  FROM ( SELECT sum(col1) as sum_col1
              , DECODE( sum(col1), NULL, 0 ) as decode_sum_col1
           FROM t1
          GROUP BY col2
       ) v1;
```

在v1中decode_sum_col1因为不在父块中使用被pruning 但是此时为DECODE的argument的sum(col1)也存在pruning的问题 但是sum(col1)已经在select list中指定并在view父块中使用因此不能删除

修改前应对

无

ISSUE-5336 在gpec的SELECT INTO语句中使用SUBQUERY则不能用SELECT INTO语句处理

概要

如果gpec对SELECT INTO语句后面有SUBQUERY的SQL进行解析则无法用SELECT INTO语句进行处理

现象及症状

以下是在SELECT INTO子句使用主机变量排序后使用SUBQUERY的示例

```
EXEC SQL BEGIN DECLARE SECTION;
int no[10];
int count[10];
EXEC SQL END DECLARE SECTION;


EXEC SQL SELECT empno, B.COUNT
    INTO :no, :count,
```

```
    FROM emp, (SELECT count(*) as COUNT FROM emp);
```

若执行上述示例则会如下报错

```
SQLCODE :-16289

SQLSTATE: 42000

ERROR MSG : into clause can have only one row
```

修改前应对

在SELECT INTO语句中对cursor进行fetch而不使用主机变量排序

ISSUE-5277 删除磁盘表空间后无法重新使用buffer发生hang

概要

如果磁盘表空间被删除缓存删除ager thread的表空间页面的buffer cache将转移到free list上 此时对于dirty页面只表示需要discard 但是如果未发生check point则无法重新使用discard页面因此存在在查找free buffer的会话中发生hang的问题对该问题进行了修改

现象及症状

如果缓存被删除的磁盘表空间页面的buffer为dirty页面则在访问磁盘表的会话中找不到free buffer从而产生hang

修改前应对

执行check point清理已删除的磁盘表空间的dirty页面

ISSUE-4945 为运行CYCLONECLUSTONECYMON在创建表的过程中使用SQLTables确认表的存在与否

概要

修改使得通过SQLTables确认是否存在运行CYCLONECLUSTONECYMON所需的表后，创建表

现象及症状

无

修改前应对

进行prepare时执行有关table exist的validate后用其结果判断表的存在与否

ISSUE-5218 如果使用async commit在一个会话中事务可能会被过度使用

概要

如果在cluster环境中使用async commit即使在事务尚未结束的情况下也可以使用新的事务，因此在一个会话中可以使用两个以上的事务

现象及症状

过度使用事务可能会发生事务不足的现象

修改前应对

无

ISSUE-5029 在cluster环境下Cyclone运行过程中即使master通过reset all选项进行了re-join也不会初始化现有的复用信息

概要

在cluster环境中Cyclone正在运行的状态下如果使用reset all选项重新启动现有的master，则应该不使用现有的复用信息而是从现在开始重新进行复用

现象及症状

即使使用reset all选项重新启动了master也使用现有的复用运行信息执行recovery

修改前应对

无

V5.0 22.1 Patch Notes

ISSUE-5132 在Cluster环境下支持对没有global secondary index的single domain表执行DML

概要

在Cluster环境下对具有一个domain的表进行DML时如果不构成global secondary index可能会失败 在DML查询中要求global secondary index是为了保障server之间的数据一致性 因此,如果仅在一个server上装载数据无需保障server之间的数据一致性则无需global secondary index即可支持DML

在Cluster环境下要想在多个server上管理一个表建议构建global secondary index

现象及症状

```
--# G4 group中仅存在G4N1
CREATE TABLE r ( c1 INTEGER )
   CLONED
   AT CLUSTER GROUP g4
   WITHOUT GLOBAL SECONDARY INDEX;

--# result: success
INSERT INTO r VALUES (1), (2), (3);

--# 改善事项
```

```
--# result: success
DELETE FROM r WHERE c1 = 2;


ERR-42000(16519): global secondary index expected in cluster DML
```

修改前应对

无

SSUE-5193 在Cluster环境下进行包括index backward scan在内的查询时结果会出现错误

概要

在Cluster环境下进行用户查询时必须访问remote服务器如果ORDER BY语句或hint包含index backward scan查询结果可能会按index forward scan的顺序出现

现象及症状

在Cluster环境下进行用户查询时如果需要访问remote服务器则构成generated query传达给remote服务器 在构成包括Index backward scan在内的generated query时access path hint信息被错误构成导致在remote服务器上执行index forward scan的情况发生

```
CREATE TABLE t1
(
    c1 INTEGER
)
```

```
    SHARDING BY RANGE ( c1 )
    SHARD s1 VALUES LESS THAN ( 10 )       AT CLUSTER GROUP g1,
    SHARD s2 VALUES LESS THAN ( MAXVALUE ) AT CLUSTER GROUP g2;

INSERT INTO t1 VALUES ( 12 );
INSERT INTO t1 VALUES ( 11 );
INSERT INTO t1 VALUES ( NULL );

CREATE INDEX t1_idx1 ON t1( c1 ASC NULLS LAST );



--# BUGBUG
--# result: 3 rows
--#         null
--#           12
--#           11
\EXPLAIN PLAN
SELECT c1 FROM t1@g2 ORDER BY c1 DESC NULLS FIRST;

  C1
----
  11
  12
null
```

```
3 rows selected.
```

修改前应对

无

ISSUE-5109修改了Shard重新配置或split brain后Cyclone故障的问题

概要

重新配置Shard或在split brain情况后进行恢复时Cyclone出现"internal error occurred(Not Need Rebalance)"错误而终止

现象及症状

在Cyclone判断不需要rebalance的情况下也有执行rebalance的情况这种情况在shard重新部署或split brain情况下可能会发生

修改前应对

以--reset重新启动Cyclone

ISSUE-5162 修改了即使设置了ODBC的SQL_ATTR_CONNECTION_TIMEOUT，也会长时间等待的问题

概要

修改了为了识别网络中断设置了ODBC的SQL_ATTR_CONNECTION_TIMEOUT但比给定的TIMEOUT等待时间更长的问题

现象及症状

虽然设置了ODBC的SQL_ATTR_CONNECTION_TIMEOUT但在特定情况下由于无法识别网络中断ODBC存在继续等待服务器响应的问题

修改前应对

在odbc.ini中添加以下属性以便快速识别网络中断

- KEEPALIVE_IDLE_TIME
- KEEPALIVE_INTERVAL
- KEEPALIVE_COUNT

或者通过改变如下内核属性快速识别网络中断

- net.ipv4.tcp_keepalive_intvl
- net.ipv4.tcp_keepalive_probes
- net.ipv4.tcp_keepalive_time
- net.ipv4.tcp_retries2

ISSUE-5160 修改了ODBC global connection环境中存在LONGVARCHAR, LONGVARBINARY参数时 客户端内存增加的问题

概要

修改了在ODBC global connection环境下反复执行带有LONGVARCHARLONGVARBINARY参数的statement时客户端内存增加的问题

现象及症状

在ODBC global connection环境下以相同的statement反复执行具有LONGVARCHARLONGVARBINARY参数的SQL时客户端内存会增加

修改前应对

无

ISSUE-5156 更改了部分错误的SQLSTATE

概要

更改了部分错误的SQLSTATE

现象及症状

被更改的错误的SQLSTATE如下

错误编号	原 SQLSTATE	更改后的 SQLSTATE	信息
13034	RD000	08S01	Service is not available
16351	08000	HY000	failed to connect to the cluster member '%s'
16523	HY000	08S01	he database system is shutting down
25001	HY000	08001	Server is not running

修改前应对

无

ISSUE-5147 修改了在CYMON的环境设置中即使设置PROTOCOL=TCP也通过DA连接的问题

概要

修改了在CYMON的环境设置中即使设置PROTOCOL=TCP也通过DA连接的问题改为通过TCP连接

现象及症状

在CYMON环境设置文件中设置PROTOCOL=TCP时应使用TCP连接但却是通过DA连接的

修改前应对

无

ISSUE-5004 在集群环境通过cyclone进行同步的过程中slave的pre-process阶段发生deadlock

概要

在集群环境通过cyclone进行同步的过程中可能会间隙性地发生deadlock

现象及症状

不再进行同步并且出现似乎停止的现象这是在集群环境中为同步处理而进行pre-process过程中发生deadlock的情况即使通过cymon进行监控也不会再进行同步

修改前应对

重设cyclone master及slave

ISSUE-4985 在CYCLONE的监控信息中添加了slave的执行信息

概要

在CYCLONE的监控信息中添加了slave处理中的信息(Apply_FileSeq, Apply_BlockSeq, Apply_Commit_Lsn)

现象及症状

无

修改前应对

无

ISSUE-4882 CYCLONE SYNC处理中发生错误时将错误详细化以便报告

概要

若在CYCLONE的SYNC处理过程中发生错误则将*ERROR OCCURRED* 信息和错误信息一起详细记录到trace log

现象及症状

无

修改前应对

无